STOCHASTIC MODELLING AND MULTIFRACTAL CHARACTERIZATION OF DIELECTRIC DISCHARGES USING LAPLACIAN FRACTALS

by Lawrence H. Arendt, P.Eng.

A thesis submitted to the Faculty of Graduate Studies
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg, Manitoba, Canada

Thesis Advisor: W. Kinsner, Ph.D, P.Eng.

ISBN 0-612-16082-3

Name _____

Dissertation Abstracts International and *Masters Abstracts International* are arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation or thesis. Enter the corresponding four-digit code in the spaces provided.

Electrical Engineering
SUBJECT TERM

0544
SUBJECT CODE

UMI

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS
Architecture 0729
Art History 0377
Cinema 0900
Dance 0378
Fine Arts 0357
Information Science 0723
Journalism 0391
Library Science 0399
Mass Communications 0708
Music 0413
Speech Communication 0459
Theater 0465

EDUCATION
General 0515
Administration 0514
Adult and Continuing 0516
Agricultural 0517
Art 0273
Bilingual and Multicultural 0282
Business 0688
Community College 0275
Curriculum and Instruction 0727
Early Childhood 0518
Elementary 0524
Finance 0277
Guidance and Counseling 0519
Health 0680
Higher 0745
History of 0520
Home Economics 0278
Industrial 0521
Language and Literature 0279
Mathematics 0280
Music 0522
Philosophy of 0998
Physical 0523
Psychology 0525
Reading 0535
Religious 0527
Sciences 0714
Secondary 0533
Social Sciences 0534
Sociology of 0340
Special 0529
Teacher Training 0530
Technology 0710
Tests and Measurements 0288
Vocational 0747

LANGUAGE, LITERATURE AND LINGUISTICS
Language
- General 0679
- Ancient 0289
- Linguistics 0290
- Modern 0291

Literature
- General 0401
- Classical 0294
- Comparative 0295
- Medieval 0297
- Modern 0298
- African 0316
- American 0591
- Asian 0305
- Canadian (English) 0352
- Canadian (French) 0355
- English 0593
- Germanic 0311
- Latin American 0312
- Middle Eastern 0315
- Romance 0313
- Slavic and East European 0314

PHILOSOPHY, RELIGION AND THEOLOGY
Philosophy 0422
Religion
- General 0318
- Biblical Studies 0321
- Clergy 0319
- History of 0320
- Philosophy of 0322

Theology 0469

SOCIAL SCIENCES
American Studies 0323
Anthropology
- Archaeology 0324
- Cultural 0326
- Physical 0327

Business Administration
- General 0310
- Accounting 0272
- Banking 0770
- Management 0454
- Marketing 0338

Canadian Studies 0385
Economics
- General 0501
- Agricultural 0503
- Commerce-Business 0505
- Finance 0508
- History 0509
- Labor 0510
- Theory 0511

Folklore 0358
Geography 0366
Gerontology 0351
History
- General 0578
- Ancient 0579
- Medieval 0581
- Modern 0582
- Black 0328
- African 0331
- Asia, Australia and Oceania 0332
- Canadian 0334
- European 0335
- Latin American 0336
- Middle Eastern 0333
- United States 0337

History of Science 0585
Law 0398
Political Science
- General 0615
- International Law and Relations 0616
- Public Administration 0617

Recreation 0814
Social Work 0452
Sociology
- General 0626
- Criminology and Penology ... 0627
- Demography 0938
- Ethnic and Racial Studies 0631
- Individual and Family Studies 0628
- Industrial and Labor Relations 0629
- Public and Social Welfare 0630
- Social Structure and Development 0700
- Theory and Methods 0344

Transportation 0709
Urban and Regional Planning 0999
Women's Studies 0453

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES
Agriculture
- General 0473
- Agronomy 0285
- Animal Culture and Nutrition 0475
- Animal Pathology 0476
- Food Science and Technology 0359
- Forestry and Wildlife 0478
- Plant Culture 0479
- Plant Pathology 0480
- Plant Physiology 0817
- Range Management 0777
- Wood Technology 0746

Biology
- General 0306
- Anatomy 0287
- Biostatistics 0308
- Botany 0309
- Cell 0379
- Ecology 0329
- Entomology 0353
- Genetics 0369
- Limnology 0793
- Microbiology 0410
- Molecular 0307
- Neuroscience 0317
- Oceanography 0416
- Physiology 0433
- Radiation 0821
- Veterinary Science 0778
- Zoology 0472

Biophysics
- General 0786
- Medical 0760

EARTH SCIENCES
Biogeochemistry 0425
Geochemistry 0996
Geodesy 0370
Geology 0372
Geophysics 0373
Hydrology 0388
Mineralogy 0411
Paleobotany 0345
Paleoecology 0426
Paleontology 0418
Paleozoology 0985
Palynology 0427
Physical Geography 0368
Physical Oceanography 0415

HEALTH AND ENVIRONMENTAL SCIENCES
Environmental Sciences 0768
Health Sciences
- General 0566
- Audiology 0300
- Chemotherapy 0992
- Dentistry 0567
- Education 0350
- Hospital Management 0769
- Human Development 0758
- Immunology 0982
- Medicine and Surgery 0564
- Mental Health 0347
- Nursing 0569
- Nutrition 0570
- Obstetrics and Gynecology .. 0380
- Occupational Health and Therapy 0354
- Ophthalmology 0381
- Pathology 0571
- Pharmacology 0419
- Pharmacy 0572
- Physical Therapy 0382
- Public Health 0573
- Radiology 0574
- Recreation 0575
- Speech Pathology 0460
- Toxicology 0383

Home Economics 0386

PHYSICAL SCIENCES
Pure Sciences
Chemistry
- General 0485
- Agricultural 0749
- Analytical 0486
- Biochemistry 0487
- Inorganic 0488
- Nuclear 0738
- Organic 0490
- Pharmaceutical 0491
- Physical 0494
- Polymer 0495
- Radiation 0754

Mathematics 0405
Physics
- General 0605
- Acoustics 0986
- Astronomy and Astrophysics 0606
- Atmospheric Science 0608
- Atomic 0748
- Electronics and Electricity 0607
- Elementary Particles and High Energy 0798
- Fluid and Plasma 0759
- Molecular 0609
- Nuclear 0610
- Optics 0752
- Radiation 0756
- Solid State 0611

Statistics 0463

Applied Sciences
Applied Mechanics 0346
Computer Science 0984
Engineering
- General 0537
- Aerospace 0538
- Agricultural 0539
- Automotive 0540
- Biomedical 0541
- Chemical 0542
- Civil 0543
- Electronics and Electrical 0544
- Heat and Thermodynamics ... 0348
- Hydraulic 0545
- Industrial 0546
- Marine 0547
- Materials Science 0794
- Mechanical 0548
- Metallurgy 0743
- Mining 0551
- Nuclear 0552
- Packaging 0549
- Petroleum 0765
- Sanitary and Municipal 0554
- System Science 0790

Geotechnology 0428
Operations Research 0796
Plastics Technology 0795
Textile Technology 0994

PSYCHOLOGY
General 0621
Behavioral 0384
Clinical 0622
Developmental 0620
Experimental 0623
Industrial 0624
Personality 0625
Physiological 0989
Psychobiology 0349
Psychometrics 0632
Social 0451

THE UNIVERSITY OF MANITOBA

FACULTY OF GRADUATE STUDIES

COPYRIGHT PERMISSION

STOCHASTIC MODELLING AND MULTIFRACTAL CHARACTERIZATION OF DIELECTRIC DISCHARGES USING LAPLACIAN FRACTALS

BY

LAWRENCE H. ARENDT

A Thesis/Practicum submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

ABSTRACT

Dielectric breakdown due to impressed electric fields upon dielectrics is a naturally occurring phenomenon within air gaps and within solid dielectrics. The resultant random discharge patterns (or trees) cannot be modeled and simulated using standard analytical methods. Instead, stochastic fractal models which require the solution of Laplace's equation of the potential field must be employed to simulate and to "grow" similar discharge tree structures.

This thesis presents the implementation and results of research undertaken a) to increase the speed of convergence of iterative methods employed to solve the finite difference approximation to Laplace's equation, b) to improve the accuracy of the solution, and c) to measure and to characterize the multifractality of the resultant Laplacian fractals.

Previous methods used only point Jacobi or point Gauss-Seidel iterative techniques to solve the finite difference approximation to Laplace's equation. This thesis extends the research to include accelerated *point iterative methods*, *line iterative methods*, *block iterative methods*. Due to the moving boundary problem caused by the growing discharge, the eigenvalues and the spectral radius of the equivalent iteration matrix change at each growth step, complicating the estimation of the optimal acceleration factor. A new method of estimating the optimal acceleration factor based on segmentation of the discrete mesh into singular-free regions, is presented. It reduces the average residual error of the mesh potentials to 25% of the level achieved by classical residual norm-based estimates of the acceleration factor.

The *multifractality* of the generated fractal discharge trees was measured and analysed using the generalized Rényi dimensions and the singularity spectrum, also known as the Mandelbrot dimension. The self-similarity multifractality is drastically impacted by the colour of random generator used to model the stochastic nature of the discharge. These measures are relatively unaffected by the selection of accelerated or non-accelerated iterative method used to solve for the potential, for which the self-similarity dimension lies in the interval of 1.636 to 1.68.

ACKNOWLEDGEMENTS

The author would like to express his appreciation and gratitude to Prof. W. Kinsner for offering the *Fractals and Chaos Engineering* course, for all the time and effort he spent in preparing the excellent course notes and related technical reports [Kins94a,,,f], for suggesting the initial Laplacian fractals course project [Aren94], and for encouraging this continuing research. Thanks for your work in proposing this project to the Centre. Thanks for being available, for explaining, for guiding, and for listening. What more can I say?

Thanks to Hongjin Chen for the collaboration and assistance with understanding some of the mathematics. Thanks to all the fellow members of the Delta Signal and Image Compression Group at the University of Manitoba for their interest and support.

Special thanks to Dennis Woodford for believing in and promoting this research, and to the board of the Manitoba HVDC Research Centre who approved this research as a Centre project. This research was carried out under the auspices and full support from the Manitoba HVDC Research Centre, project number E4.194. Use of the Centre's computing facilities to perform simulation runs and to print out this thesis is acknowledged. It is hoped that the Centre can further exploit this technology with the objective to improving the design of dielectrics. NSERC is also acknowledged.

The author would like to express his gratitude to all the other people who contributed in any way to this work or who actively encouraged this work.

To my wonderful friend and wife Katharina, who has been very understanding and supportive; without the extra love and patience she manifested to her husband and their children, this would not have been possible, "Thank you so much! ". To the author's 27 month old son Victor who wants "Daddy down[stairs]", to play "Puppy, Rabba, run fa-a-st !" and "Puppy, Rabba hi-i-ding !!!", I say "Victor, put on your runners, I'm almost finished!".

This thesis is dedicated to my very loving and supportive family. Without their prayers, this would not have been possible.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

386	an Intel, AMD, or TI 80386-based personal computer.
486	an Intel, AMD, or TI 80486-based personal computer.
ADI	alternating direction implicit. Implementation as per Peaceman-Rachford, Douglas-Rachford, Douglas, or similar, having multiple optimal parameters or multiple, sequenced acceleration factors.
ADSBOR	alternating direction, successive block over-relaxation (single acceleration factor).
ADSLOR	alternating direction, successive line over-relaxation (single acceleration factor).
*.bmp	Microsoft bitmap file.
C	the "C" programming language.
C++	the "C++" object-orientated programming language, an enhancement of "C".
*.cdf	a corona data file.
cpu	central processing unit, here it generally refers to a 486 or Pentium processor.
DBM	dielectric breakdown model.
DLA	diffusion-limited aggregate.
G-S, or GS	Gauss-Seidel iterative technique.
HDD	hard disk drive.
JBOR	Jacobi block over-relaxation
*.pnt	a point file containing only coordinates of points, usually corona points.
*.rbf	a runtime batch file.
SOR	successive (point) over-relaxation.
SBOR	successive block over-relaxation.
SLOR	successive line over-relaxation.
SSOR	symmetric successive over-relaxation.
SVGA	super video graphics adapter.
vel	volume element

LIST OF SYMBOLS AND VARIABLES

α	scaling index.
$\mathbf{A}$	matrix of coefficient of unknown potentials **u**.
$\mathbf{b}$	column vector containing the knowns in linear equations for U.
δ	pseudo-residual vector or displacement vector.
D_C	correlation dimension.
D_H	Hausdorff dimension.
D_{HM}	Hausdorff mesh (counting) dimension.
D_I	information dimension.
D_q	generalized (Rényi) dimension.
D_S	self-similarity dimension.
$\mathbf{e}$	error vector.
$\in$	permittivity.
ϕ	continuous potential.
$f(\alpha)$	spectrum of scaling indices, singularity or multifractal spectrum, or the Mandelbrot dimension.
$\mathbf{G}$	iteration matrix.
h	mesh spacing.
i,j	coordinates of any mesh point P.
k	iteration step, computation stage, block size.
λ_{max}	spectral radius of G, maximum or dominant eigenvalue of **G**.
$\hat{\lambda}_{max}$	estimate of the spectral radius of **G**.
n	number of mesh points.
N_k	number of vels intersected by fractal F at the k^{th} stage of computation, or the number of points on the fractal F.
N_r	number of vels of size r intersected by fractal F.
$\mathbf{O}$	asymptotic rate.
p	norm number.
q	dimension number.
r	radius or width of a vel.
$\mathbf{r}$	residual vector.
R	rate of convergence of iterative method.
$\mathbf{u}$	column vector of all unknown potentials U.
$\underline{\mathbf{u}}$	exact values of **u**.
$\mathbf{u}^k$	estimate of **u** at iteration step k.
$U_{i,j}$	potential at mesh point P(i,j).
U^k	set of all mesh potentials at iteration step k.
$\mathbf{v}$	iteration column vector derived from **b**.
V	an arbitrary vector.
ω	acceleration factor.
ω_{opt}	optimal acceleration factor.

CHAPTER 1

INTRODUCTION

1.1 Problem Definition

Electrical discharges in dielectrics exhibit a stochastic behaviour in that no discharge is exactly like another. Even identical initial conditions can lead to similar but different discharge patterns. Modelling the growth of such phenomena requires the use of a stochastic model which cannot be implemented by traditional methods. The resultant discharges are generally referred to as Laplacian fractals or Laplacian multifractals since their growth process is governed by the solution of Laplace's equation for the potential field, and their structure exhibits global scaling properties best described or characterized by fractal and multifractal measures.

A common method for solving Laplace's equation for non-trivial configurations in two dimensions is to employ a finite-difference representation on a two-dimensional N×M grid (or lattice, or mesh). This leads to (N×M) linear equations, one for each grid point, which can be arranged in standard $\mathbf{Au} = \mathbf{b}$ matrix form where $\mathbf{A}$ is square matrix of size (N×M). Direct solution methods are intractable for even reasonable (N×M) such as 100×100 or even 25×25, so iterative techniques are employed.

For stationary Laplacian field problems in which the boundaries for the given system are fixed, the determination of the spectral radius of the iteration matrix, and hence the optimal acceleration factor, is a trivial matter for square or rectangular meshes, but more involved for circular meshes and other shapes [Lewi71, LIMc68]. The optimal acceleration factor is computed once prior to any iterations, and an approximation of the

solution is then obtained by successive iterations until any predetermined convergence criterion is satisfied, and the problem is considered to be solved. The rate of convergence is governed by the maximum eigenvalue, also known as the spectral radius, of the associated iteration matrix.

When we endeavour to model dielectric discharges, we have the problem of non-stationary (moving) Dirichlet boundaries in the interior of the mesh as the corona grows in discrete steps. At each new growth step, the corona advances to occupy another point on the mesh. Since the corona is set to 0 potential, the new growth site has now become another Dirichlet boundary in the interior of the mesh. This is equivalent to removing (deleting) the associated row and column in the equivalent iteration matrix, changing (in fact decreasing) the associated spectral radius. Analytically determining the new spectral radius of the reduced matrix is tantamount to directly solving the finite difference matrix; both are computationally intractable. We must therefore estimate the new spectral radius from some norms, and use that estimate to compute the instantaneous acceleration factor ω_{opt}. Since the potential field has been drastically disturbed in the locality of the new growth site, and somewhat disturbed globally, the potential field must be recalculated everywhere.

In juxtaposition to the previously mentioned stationary Laplacian field problem, modelling of dielectric discharges requires the recalculation of the field after *each* growth step. In order to achieve high resolution and reduce any discretization and edge effects, we must employ as large a mesh as possible. In addition, the accuracy of the field solution is dependant on the number of iterations. It should be obvious that the goals of achieving

the maximum accuracy on as large a grid as possible in the shortest time possible are mutually exclusive.

All of the published literature encountered to date concerning the fractality of dielectric discharges have applied one or more different dimension measures to Laplacian fractals, with each dimension measure providing a single value for the dimension, usually slightly different for each measure used. Since the fractal dimension of pure fractals is invariant of the fractal measure used, it can only be concluded that the Laplacian fractals are indeed multifractals! To properly characterize the multifractality of any multifractal object, multifractal dimension measures must be employed. It is interesting to note that there is an absence of published literature on the multifractality of Laplacian discharge fractals and one can only assume that the multifractality has not been studied. This thesis attempts to rectify that condition by applying the generalized Rényi dimension and the singularity spectrum (Mandelbrot dimension) measures to the discharge fractals generated during the course of the author's research. At present, the multifractal behaviour and characteristics of discharge fractals are not known, nor their sensitivity to computational and modelling methods.

1.2 Objectives

There are several objectives of this thesis. The first major objective is to improve the computational methods used, with the emphasis on improving the rate of convergence of solving the Laplacian field problem by using accelerated iterative methods *and* also improving the accuracy of the solution. These parameters are not mutually exclusive. To this end, non-accelerated and accelerated point, line, and block iterative methods will be

implemented and their performance assessed. Due to the moving boundary problem, the optimal acceleration factor ω_{opt} is not easily determinable, and an estimate must be derived from residual norms. The information extractable about the computation and growth processes will be increased to provide a better understanding of the numerical behaviour and convergence of the iterative methods.

The second major objective of this thesis is to extend the concepts and measures of multifractality to the dielectric discharge fractals to study and characterize their multifractality with the hopes of comparison, classification, and grouping of discharges occurring under different conditions. It is also desired to assess the sensitivity and impact of modelling and computational methods on the multifractality of the resultant discharges. The Laplacian fractals will be grown in the same homogeneous dielectric configurations as in previous research [Stace95, Aren94] for comparison purposes, with the addition of multiple treeing inside the parallel plates configuration.

1.3 Structure of Thesis

This thesis has a structure designed to match the generation and analysis of dielectric discharges. Chapter 2 contains the background necessary to appreciate the research and results of this thesis. Firstly, the dielectric breakdown model is presented. The evolution of the model and the published implementation details are examined and compared. As mentioned above, the implementation requires the solution of Laplace's equation for the potential field after each discrete stage of growth. This necessitates the discussion of direct and iterative methods, the handling of moving boundaries, and the acceleration of convergence of iterative methods. The non-accelerated and accelerated

point, line and block iterative methods employed to solve the finite difference approximation to Laplace's equation and are presented in detail. Since it is known that Laplacian fractals are multifractals, the concepts of generalized (Rényi) fractal dimensions and multifractality (Mandelbrot dimension) are presented and emphasized. The well-known box counting dimension, which is embedded in the Rényi spectrum, receives special consideration as a single-dimension measure for comparison with published results.

Chapter 3 discusses the software programs and object classes developed to simulate dielectric discharges, extract information about the growth process and convergence behaviour, and measure their multifractality.

The results of experiments are discussed in Chapter 4. The experiments were designed to assess the sensitivity of the resultant fractal discharges to parameter variations, to compare the discharges with those from previous research, and to validate and calibrate the fractal and multifractal dimension measures. The limitations of the implementation methodology are included here as well.

This is followed by conclusions and recommendations for further research in Chapter 5.

CHAPTER 2

BACKGROUND

2.1. Introduction

This chapter presents the material that is required to understand and appreciate the research for this thesis. It begins with a discussion of the physical processes involved in the breakdown of dielectrics which leads to the occurance of discharges. Although the process is generally understood, it is impossible to mathematically model and solve using standard analytical methods. Instead, a generalized, stochastic dielectric breakdown model must be used. This model and its limitations are then discussed. This is followed by an in-depth discussion of practical numerical methods available to solve Laplace's equation for the potential field. Different iterative methods and the determination of convergence of such iterative techniques are discussed. Finally, fractals, multifractality, and the measures of the multifractality are introduced and presented.

2.2. The Dielectric Breakdown Process

Dielectric breakdown discharges are a complex yet common-place phenomenon. The most familiar, (audibly and) visibly-impressive example identifiable by nearly everyone is the phenomenon of lightning. On a microscopic scale, the failure of semiconductors due to "invisible, silent" static electricity is another example. Other examples include surface discharges known as Lichtenberg figures, the treeing in polymers, and the breakdown of capacitor and cable dielectrics.

Dielectric breakdown is caused by the presence of a high intensity electric field impressed upon some dielectric material which causes some molecules of the dielectric to become ionized and conduct current. In the case of gaseous dielectrics, dielectric breakdown results in an ionized conduction channel that grows from one electrode to another. This channel may be continuous or self-extinguishing leading to repetitive discharges. The damage to the dielectric may or may not be permanent. In the case of solid dielectrics the discharge permanently damages the dielectric material, leaving behind trees (or tracks) in the material, and compromising the insulating properties of the material. Dielectric breakdown discharges can cause catastrophic component failures evidenced by physical cracking of components, as has been observed in the failure of high-voltage wall bushings at the Dorsey Converter station operated by Manitoba Hydro.

Wiesmann and Pietronero separate the breakdown process into two phases: a non-conducting or dielectric/insulating phase P_d, and a conducting phase P_c [WiPi86]. Kinsner classifies the breakdown process into four phases: a non-conducting phase P_n (same as P_d), a growth transition phase P_t (same as P_c), a growth steady-state phase P_s, and a possible punch-through phase P_p [KiSt94].

On a microscopic level, each bond within the dielectric material will initially experience a non-conducting phase, and possibly the conducting phase, depending on the activity of the corona-dielectric interface. If an electric field $\mathbf{E}_c$ exceeding a critical value is impressed upon the dielectric, the corona discharge will grow by "breaking" a bond along the interface, this involving the creation of charge, transportation of charge across the bond to the new site molecule, and annihilating the previously existing site charge. This sequence of events occurs within the conducting phase, and upon the completion of

the charge activity and transportation, the site and the connecting bond, now part of the corona discharge, returns to a non-conducting phase. Thus the breakdown process, which consists of a non-conducting and conducting phase, is repeated each time a bond is broken. The method by which certain bonds are selected for breakdown is discussed later. The discharge as a whole will eventually reach either the punch-through or steady-state phase.

2.2.1 Conducting Phase

In the conducting phase, the relationship between the charge density ρ, total charge Q, dielectric permittivity ε and electrostatic potential ϕ is governed by the following equations [WiPi86]

$$\nabla^2\phi = -\frac{\rho}{\varepsilon} \quad \text{(Poisson's equation)} \tag{2.1}$$

$$Q = \sum_{s,k} e\sigma\, n_k(\sigma) \tag{2.2}$$

where

$n_k(\sigma)$ designates the non-linear density of the charge carriers with charge ($e\sigma$),

e is the electron charge,

σ is an infinitesimal volume, and

k classifies different categories of carriers.

For the dielectric material, when the current to the new discharge site is extinquished, then

$$n_k(\sigma) = 0 \tag{2.3}$$

The *transport equation* for the creation, transportation, and annihilation of charge

due to field and diffusion currents during the conducting phase is

$$\frac{\partial n_k}{\partial t} = D_k(\sigma) \cdot n_k(\sigma) \tag{2.4}$$

where $D_k(\sigma)$ is a non-linear, local operator such that

$$D_k(\sigma) = f\left(\mathrm{E}, n_k(\sigma)\right) \tag{2.5}$$

The electrostatic field **E,** excluding photoionization, is related to the electrostatic potential by

$$\nabla\phi = -\mathrm{E} \tag{2.6}$$

2.2.2 Steady State and Punch-Through Phases

The growth steady-state phase occurs when a steady-state equilibrium is achieved, and the expansion of the interface is essentially stalled or arrested. Punch-through occurs when the corona discharge completely traverses the dielectric and reaches the other conductor. This is analogous to a destructive short-circuit spark between two conductors. Both phases are still governed by Eqs. 2.1 through 2.5.

2.2.3 Non-Conducting Phase

Since the charge density in the non-conducting phase is zero, Eq. 2.1 becomes

$$\varepsilon\nabla^2\phi = 0 \tag{2.7}$$

which for homogeneous dielectrics (ε is constant) reduces to the Laplace equation

$$\nabla^2\phi = 0 \tag{2.8}$$

and the electric field at any site can be determined from

$$\nabla\phi = -\mathrm{E} \tag{2.6}$$

It should be noted here, that none of the above equations contain a term which describes the stochastic nature of the dielectric breakdown process.

2.3. The Generalized Dielectric Breakdown Model

Although the phenomenon of dielectric breakdown is generally understood, it is difficult if not impossible, to mathematically model and solve using analytical methods. This is due to the non-linear coupled system of the *Poisson/Laplace* equation (Eq. 2.1) governing the field distribution and the *transport* equation (Eq. 2.4) governing the charge activity which must be solved for the 2 or 3 dimensional object of interest. In addition, dielectric breakdown is an example of a *stochastic* growth process, and its randomness cannot be described by any equation.

It appears that development of Laplacian fractals derived from the study of diffusion-limited aggregates or DLA's [WiSa81] and attempts to refine the DLA model [PeJS92, p. 479]. The first major attempt to model dielectric discharges came from Sawada et al [SOYH82]. In their model, a neighbouring site adjacent to the existing corona discharge was randomly chosen. The directional growth of corona was controlled by an *apriori* tip (growth) priority factor R (R = 1 to 200) where

$$R = \frac{p_{tip\ growth}}{p_{new\ branch\ growth}} \tag{2.9}$$

This was intended to favour treeing; instead the resultant pattern resembled the space filling eden (or cancer) models. This is not surprising since the model attempted to control a stochastic character by a deterministic element, and secondly, their model totally ignored any concept of, and linkage to, electric field intensity effects.

2.3.1 The Standard Model

The accepted standard reference model for modelling dielectric discharges was developed and introduced by Pietronero, Wiesmann and Niemeyer [NiPW84, PiEW86, WiPi86] and has become the basis for further research [Satp86, FuSa87, Ever90, Aren94, KiSt94, Stac95, to name a few]. Since the breakdown process is complex, assumptions must be made and some effects are ignored in order to have a useable *generalized* dielectric breakdown model (DBM).

The model requires a discrete lattice (or mesh) - based representation to approximate the continuous dielectric material and electrode configuration, and the continuous potential field. The model assumes some arrangement of a high potential electrode separated from a zero potential electrode by a homogeneous dielectric material. This thesis studied only the parallel plate capacitor-type dielectric model. To simplify the ideal model, the corona is assumed to be perfectly conducting and having a potential of zero, and the conduction phase described by Eq. 2.4 is assumed to be perfectly conducting [WIPi86], and all electrostatic and magnetostatic effects due to the flowing charges are ignored. Additionally, we assume the charges to have reached equilibrium prior to the non-conducting phase. Having made these gross assumptions and neglecting any enclosed charge, we can employ Laplace's equation to approximate the potential at any point in the dielectric.

The interface between the corona and the dielectric consists of all possible growth sites which are the nearest-neighbour sites to the corona. Each of these possible growth sites (or candidate sites) is connected to the corona by at least one unbroken bond, these bonds being referred to as *candidate bonds*.

The analysis of the physics that determines *where the discharge begins* is very complex, so for this thesis, we assume a random, arbitrary starting point for the corona to begin. For simplicity, we select a dielectric point adjacent to the centre of the bottom (grounded) electrode plate. The discharge starts from somewhere on the zero potential object, and grows by breaking down, one at a time, a nearest-neighbour candidate bond along the interface. At any stage k in the growth we have a set of candidate bonds B^k in which any bond B_h^k connects a candidate dielectric site D_h^k with potential ϕ_h to a corona site C_m^k which has a potential $\phi_m = 0$. D^k is the set of all candidate dielectric sites at stage k, and C^k is the set of all corona sites at stage k. Expressed in mathematical terms we have

$$B^k = \left\{(m,h) \middle| |m-h| = 1, m \in C^k, h \in D^k\right\} \tag{2.10}$$

The probability of breakdown of bond (m,h) is proportional to some function f of the electric field across the bonds such that

$$P(m,h) = \frac{f(\phi_m - \phi_h)}{\sum_{(m,h)\in B} f(\phi_m - \phi_h)} \tag{2.11}$$

where the summation is performed over all candidate bonds (m,h) ε B^k. Niemeyer et al [NiPW84] suggest a function which enhances the electric field by some exponent η. Since $\phi_m = 0$, Eq. 2.11 reduces to

$$P(m,h) = \frac{(0-\phi_h)^\eta}{\sum_{(m,h)\in B}(0-\phi_h)^\eta} = \frac{\phi_h{}^\eta}{\sum_{(m,h)\in B}\phi_h{}^\eta} \tag{2.12}$$

Therefore the growth probability of a site h is the sum of the probabilities of *any one of that site's candidate bonds* breaking down. Fukai and Sano [FuSa87] defined the probability P(i,k) of growth at site (i, k) as

$$P(i,k)=\frac{\phi_{i,k}{}^{\eta}}{\sum \phi_{i,k}{}^{\eta}} \quad \text{, for all candidate sites } (i, j). \tag{2.13}$$

The philosophical differences and the effects of using bond breakdown probabilities or site growth probabilities have been addressed elsewhere [Aren94, Stac95].

The exponent η defines the growth complexity. Although there is no physical interpretation of η, its value has a high impact on the resulting patterns and their fractal dimension. The density and self-similarity dimension of the discharge structures increase as η decreases. For $\eta = 0$, the growth probability is independent of the electric field and is equal for all growth sites, resulting in the space filling eden (cancer) model with a high degree of ramification. For η = 1, the DLA model is recovered. As η increases, the ramifications decrease, and the fractal dimension decreases. This effect of η has been discussed in previous literature [NiPW84, WiPi86, Satp86, and Ever90]. The dielectric breakdown model implemented by this thesis will adhere to the NiPW84 model.

The selection of the *next* breakdown site involves the generation of a random number R normalized to the sum of all growth probabilities, then re-summing all the bond growth probabilities and selecting the first bond which causes the sum to exceed R. Alternatively, the random number R can be normalized to the largest growth probability, and then selecting the first bond whose growth probability exceeds R. Pietronero, Wiesmann and Niemeyer have not detailed their selection process, while other researchers have [FuSa87, Aren94, Stac95]. The breakdown selection process is a fundamental issue (beyond the scope of this thesis) and caution is urged when implementing new methods.

For this research, pseudo-random number generators and chaotic generators were used to generate the sequence of random numbers. Appendix B contains a brief study of the properties and characteristics of nine pseudo-random generators and seven chaotic generators which were selected for implementation. Experimental work shows that the selection of the random generator to be used in conjunction with the dielectric breakdown model has a significant impact on the appearance, structure, and multifractality of the resultant Laplacian fractals.

When a bond breaks, the potential and electric fields are perturbed drastically in the immediate locality of the breakdown, and perturbed globally but to a lesser degree. Point iterative Jacobi and Gauss-Seidel methods are used to iteratively solve the finite difference form of Laplace's equation for the potential [KiSt94, Aren94]. Several researchers [FuSa87, KiSt94, Aren94] have increased the convergence rate by performing many local iterations within a small square local region enclosing the locality of the breakdown, followed by a few global iterations.

The process of growth followed by the solution of the disturbed field continues until the steady state equilibrium phase or the punch-through phase has been reached. The resultant structures or patterns are called *Laplacian fractals* because they have fractal properties and their dynamic growth probability is governed by the solution of Laplace's equation for the potential [WiPi86 p.154, KiSt94].

2.3.2 Model Limitations

It is known that electrical discharges are dominated by birth processes which lead to rapid corona growth. Death processes such as the decay and extinction of current and dynamic branch "wandering" or shifting occur in reality, but are not yet fully understood

and appear to have only minor impact and are therefore ignored in all the DBMs. A comprehensive understanding of these phenomenon may be unlocked by very high speed photography of actual discharges.

Stacey [Stac95] considered the simulation to be in the steady state phase when the sum of gradients exceeded the maximum potential. This seemed to stop the simulations a few steps prior to complete punch-through to the high potential electrode. This is an artificial criterion since there is no corresponding physical process or interpretation to the comparison of a sum of electric fields to a potential! All the current model implementations assume that punch-through occurs; there have not been any proposals to date on how to determine if the steady-state equilibrium phase has been attained.

The extension of the DBM to include inhomogeneous and non-linear dielectrics has not been addressed. No models to date address the property possessed by most materials in that breakdown only occurs if the electric field exceeds some critical value.

When perusing the available published literature on dielectric breakdown, it was quickly realized that not all authors presented all the details of their dielectric breakdown models. This has made it difficult to verify their results. For example, the choice of a random or chaotic generator will influence the resultant patterns, and possibly their fractal dimension. The details of the model implemented during the course of the research for this thesis will be presented later in Chapter 4 with the complete code listings appearing in Appendix H [Aren96].

2.4. Computational Methods For Laplace's Equation

Simple problems requiring the solution of Laplace's equation for some shapes can be solved using well-known analytical techniques such as

- direct integration applicable only to one-dimensional problems such as $\frac{d^2U}{dx^2}=0$,
- the product solution written in terms of infinite power series, exponentials, hyperbolic or sin/cos functions [Hayt81, p. 211], or
- possibly other techniques.

Analytical techniques are intractable, however, if the shape of the system is complex or when we introduce arbitrary non-stationary, or even stationary, boundaries into the interior of the physical system that we are trying to solve. Fortunately we can resort to a numerical technique such as finite-differencing to obtain a good approximation to the exact continuous solution.

2.4.1 The Finite Difference Form of Laplace's Equation

The continuous form of Laplace's equation for two dimensions is given by

$$\nabla^2 U(x,y)=\frac{\partial^2 U}{\partial x^2}+\frac{\partial^2 U}{\partial y^2}=0 \qquad (2.14)$$

where U is the potential at any point (x,y). A discrete representation of the continuous 2-d physical system is created by employing a finite-difference representation on a two-dimensional $N\times M$ grid, lattice, or mesh, whose points are separated by Δx horizontal and Δy vertical distances (Fig. 2.1). The continuous electrodes and dielectric material are modelled by discrete representations consisting of groups of mesh points to which are associated the appropriate potentials, dielectric constants, and other properties. Next,

Laplace's equation which governs the field for all non-electrode mesh points must be discretized.

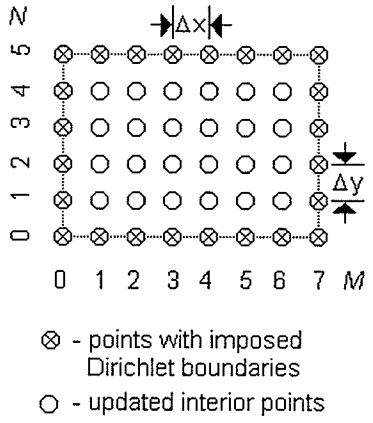


Fig. 2.1: Typical discrete mesh.

Approximating both second derivatives by centred second differences at an arbitrary point P(i,j), we obtain:

$$\left.\frac{\partial^2 U}{\partial x^2}\right|_{i,j} = \frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{\Delta x^2} + \mathrm{O}\left(\Delta x^2\right) \tag{2.15}$$

$$\left.\frac{\partial^2 U}{\partial y^2}\right|_{i,j} = \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{\Delta y^2} + \mathrm{O}\left(\Delta y^2\right) \tag{2.16}$$

Substituting Eq. 2.15 and Eq. 2.16 into Eq. 2.14, the *discrete form* of Laplace's equation can be written as

$$\nabla^2 U\Big|_{i,j} = \frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{\Delta x^2} + \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{\Delta y^2} + \mathrm{O}\left(\Delta x^2\right) + \mathrm{O}\left(\Delta y^2\right) \tag{2.17}$$

If we have a discrete *uniform* mesh, $\Delta x = \Delta y = h$, and Eq. 2.17 simplifies to

$$\nabla^2 U\Big|_{i,j} = \frac{U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1} - 4U_{i,j}}{h^2} + \mathrm{O}\left(h^2\right) = 0 \tag{2.18}$$

where $O(h^2)$ is the asymptotic notation for the truncation error of the higher-order terms of the approximation. The numerical form of the *discrete approximation* to Laplace's equation can be written as

$$\nabla^2 U\Big|_{i,j} \approx \frac{U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1} - 4U_{i,j}}{h^2} = 0 \qquad (2.19)$$

Re-arranging terms we obtain

$$U_{i,j} = \left(U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1}\right)/\,4 \qquad (2.20)$$

which is generally referred to as the 5-point operator or 5-point star. Its finite-difference equation can be represented pictorially by a 5-point computational molecule or stencil. It should be noted that Eq. 2.14 is exact everywhere, Eq. 2.18 is exact for all defined mesh points, and Eq. 2.19 is the numerical approximation, numerical form or computational form of Laplace's equation. Equation 2.20 implies that the second-order finite-difference approximation to the solution of Laplace's equation at any point P(i,j) is merely the average of its 4 nearest neighbours.

Other, higher order finite difference approximations and stencils [Bick48, Waso55, BiGu74, Coll60] such as the 9-point star with $O(h^4)$ truncation, or HODIE [LyRi78] with $O(h^6)$ exist but are beyond the scope of this thesis. Most higher order approximations involve points more than a distance h from a reference point causing problems along boundaries, and are more complex computationally, sacrificing speed for accuracy.

Thus, by replacing the Laplace partial differential equation by a system of finite-difference equations based on a discrete mesh with boundary values, numerical methods have reduced the continuous system to a linear system, which has the advantages that

- it can be applied to complex shapes,
- it can be applied to systems with non-homogeneous dielectrics, and
- it is easily implemented and solved by computers.

2.4.2 Boundary Considerations

Before proceeding to the solution of the linear system, the issue of boundaries must be discussed. In the problem that we are addressing, we consider 3 types of boundaries: *Neumann* and *Dirichlet* boundaries along the sides of the mesh, and Dirichlet boundaries, also known as *singularities*, in the interior of the mesh.

2.4.2.1 Dirichlet Boundaries

Points which have an assigned, invariant potential are known as Dirichlet boundary points. Since their potentials are known and constant, they are not influenced by the solution for neighbouring points, nor do they need to be solved for. Dirichlet boundaries will, however, influence the solution for neighbouring points.

2.4.2.2 Interior Singularities

For our problem, the corona is assumed to be perfectly conducting, having a potential of 0, and all death processes are ignored. Once an interface point becomes part of the corona, its potential is set to 0 and remains invariant. Therefore all points which become part of the corona, in fact become interior Dirichlet boundaries. One fundamental difference between Dirichlet boundary points along the edge of the mesh and those in the interior of the mesh is that the iteration method knows *apriori* about the former, but must dynamically discover and avoid solving for the latter. The analytical methods devised by

Motz [Motz47] and others [BeCr73] to handle singularities are too complex to be considered for our model problem.

2.4.2.3 Neumann Boundaries

Neumann boundaries, also known as normal derivative boundaries, pose a bit more difficulty. Neumann boundaries do not possess an invariant potential, rather their potential is permitted to float, or change with the condition that the normal derivative across the boundary is zero. This allows us to model finite sections of otherwise infinitely large shapes without incurring edge-effects penalties. In general, Neumann boundaries must satisfy

$$\frac{\partial U}{\partial n}=0 \tag{2.21}$$

where n is the normal to the boundary. For Neumann boundaries along a side of the mesh, excluding corners, Eq. 2.21 becomes

$$\frac{\partial U}{\partial x}=0 \qquad \underline{\text{or}} \qquad \frac{\partial U}{\partial y}=0 \tag{2.22}$$

Referring to Fig. 2.2, we want to compute the 5-point approximation to Laplace's equation at point 0. Point 4 is a virtual exterior point. Using centred differences, the finite-difference form of Eq. 2.22 becomes

$$\left.\frac{\partial U}{\partial x}\right|_o \approx \frac{U_4-U_2}{2h}=0 \tag{2.23}$$

which implies $U_4 = U_2$. Substituting Eq. 2.23 into Eq. 2.19, we obtain

$$\left.\nabla^2 U\right|_0 \approx \frac{U_1+2U_2+U_3-4U_0}{h^2}=0 \tag{2.24}$$

which simplifies to

$$U_0 = (U_1 + 2U_2 + U_3)/4 \tag{2.25}$$

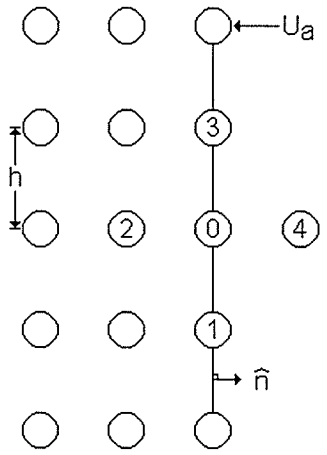


Fig. 2.2: Example of a Neumann boundary along a side.

For Neumann boundaries at a corner, Eq. 2.21 implies

$$\frac{\partial U}{\partial x} = 0 \qquad \underline{\text{and}} \qquad \frac{\partial U}{\partial y} = 0 \tag{2.26}$$

Referring to Fig. 2.3, we want to compute the 5-point approximation to Laplace's equation at point 0. Points 3 and 4 is are virtual exterior points. Using the previous method, we determine that $U_4 = U_2$ and $U_3 = U_1$, which substituting into Eq. 2.20 simplifies to

$$U_0 = (U_1 + U_2)/2 \tag{2.27}$$

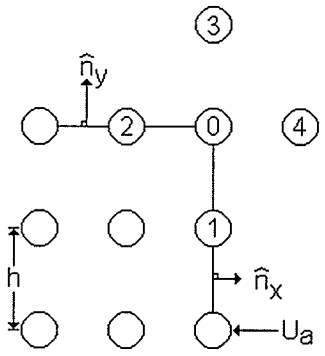


Fig. 2.3: Example of a Neumann boundary at a corner.

2.4.3 Direct Methods

The transformation from a continuous representation to a discrete representation on an $N \times M$ mesh leads to $N \times M$ linear equations, one for each grid point, which can be arranged in the standard **Au** = **b** matrix form. Here **u** contains all the unknown potentials of U arranged in column vector fashion, **A** is a square matrix with width ~ $(N \times M)$ containing the coefficients for all the linear equations of unknowns **u**, and **b** contains the Dirichlet boundaries. Thus the problem of solving

$$\nabla^2 U(x,y) = \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 0 \tag{2.14}$$

has been reduced to solving

$$\mathbf{Au} = \mathbf{b} \tag{2.28}$$

Recalling from Sec. 1.1, an objective of this thesis is to simulate the dielectric breakdown on as large a mesh as possible. The mesh dimensions considered range from 500×500 to 1000×1000 or more. It should be obvious that direct solution methods are intractable for the corresponding 250,000 and 1×10^6 square matricies **A**. This is true even for "reasonable" lattice sizes of 100×100 or 25×25. Therefore, iterative techniques must

be employed to solve for **u**.

2.4.4 Iterative Methods

2.4.4.1 Introduction

Iterative methods consist of the repeated application of a simple algorithm, but generally yield the exact solution, which would be obtained by direct methods, only as a limit of a sequence [FoWa67]. Part of the research focused on the evaluation of different iterative methods, and due to finite-time constraints, not all available methods at our disposal were evaluated. Point, line and block iterative techniques were selected since they are simple, fast, and do not require extra storage except for the point Jacobi and SER methods.

The general linear iterative solution of $\mathbf{Au} = \mathbf{b}$ (Eq. 2.28) can be described by

$$\mathbf{u}^k = \mathbf{G}\mathbf{u}^{k-1} + \mathbf{v} \tag{2.29}$$

where **G** is the iteration matrix, $\mathbf{u}^k$ is the estimate of **u** at iteration step k, and **v** is a column vector related somehow to **b** by

$$\mathbf{v} = \mathbf{M}\mathbf{b} \tag{2.30}$$

G, M, and therefore **v,** depend on the iteration method. If $\underline{\mathbf{u}}$ is the exact solution that we are after, where $\underline{\mathbf{u}} = \mathbf{A}^{-1}\mathbf{b}$, then the error vector $\mathbf{e}^k$ can be expressed as

$$\mathbf{e}^k = \mathbf{u}^k - \underline{\mathbf{u}} \tag{2.31}$$

Now $\underline{\mathbf{u}}$ is unknown, however

$$\mathbf{u}^\infty = \lim_{k\to\infty} \mathbf{u}^k = \underline{\mathbf{u}} \tag{2.32}$$

so Eq. 2.31 can be re-written as

$$\mathbf{e}^k = \mathbf{u}^k - \mathbf{u}^\infty \tag{2.33}$$

If we let k→∞, then in the limit Eq. 2.29 becomes

$$\mathbf{u}^{\infty} = \mathbf{G}\mathbf{u}^{\infty} + \mathbf{v} \tag{2.34}$$

which if we now subtract from Eq. 2.29, we obtain an expression for the error operator,

$$\mathbf{e}^{k} = \mathbf{G}\mathbf{e}^{k-1} \tag{2.35}$$

For *any* iteration method to converge, the iterative process must reduce the error so that

$$\lim_{k\to\infty} \mathbf{e}^{k} = 0 \tag{2.36}$$

for any arbitrary initial vector $\mathbf{u}^0$ which has a corresponding error of

$$\mathbf{e}^{0} = \mathbf{u}^{0} - \underline{\mathbf{u}} \tag{2.37}$$

If we now assume that **G** has n linearly independent eigenvectors v_i associated with distinct eigenvalues λ_i, then we can express $\mathbf{e}^0$ as

$$\mathbf{e}^{0} = \sum_{i=1}^{n} \alpha_i v_i \tag{2.38}$$

and by using the powers of a matrix property, $\mathbf{e}^k$ can be expressed as

$$\mathbf{e}^{k} = \sum_{i=1}^{n} \alpha_i \lambda_i^k v_i \tag{2.39}$$

If we arrange the eigenvalues in descending order so that λ_1 is the largest eigenvalue, λ_{max}, then

$$\mathbf{e}^{k} = \lambda_{\max}^{k}\left[\alpha_1 v_1 + \alpha_2\left(\frac{\lambda_2}{\lambda_{\max}}\right)^{k} v_2 + \alpha_3\left(\frac{\lambda_3}{\lambda_{\max}}\right)^{k} v_3 \ldots\right], \text{ and} \tag{2.40}$$

$$\lim_{k\to\infty}\left(\frac{\lambda_i}{\lambda_{\max}}\right)^{k} = 0, \quad \lambda_i \neq \lambda_{\max} \tag{2.41}$$

and finally,

$$\lim_{k\to\infty} \mathbf{e}^k = \lambda_{\max}^k \alpha_1 v_1 \tag{2.42}$$

The dominant (largest) eigenvalue of **G** is λ_{max}, also known as the *spectral radius* of **G**. *In order for convergence to occur, the spectral radius of the iteration process must be less than 1.0 and real.*

The rate of convergence R of Eq. 2.29 [Ames77, p. 115] for a given iteration method is related to λ_{max} by

$$R = -\log(\lambda_{max}) \tag{2.43}$$

Therefore,

a. if $\lambda_{max} > 1.0$, then $R < 0.0$, and we have divergence,

b. if $\lambda_{max} = 1.0$, then $R = 0.0$, and we have no convergence,

c. if $\lambda_{max} = 0.0$, then $R = +\infty$ and we have unrealistic instantaneous convergence!,

d. if λ_{max} is complex, then we have oscillations, and finally

e. if $0 < \lambda_{max} < 1$, then $R > 0.0$, and we have convergence.

Since we desire as great a rate of convergence as possible, we need to minimize λ_{max}. All iteration methods attempt to either reduce the number of iterations by reducing the λ_{max} associated with **G**, or to reduce the computation time required to achieve convergence, or both.

It should be noted that due to the non-stationary boundary problem, the system of linear equations that is to be solved is reduced by one equation each time the corona advances to a new mesh point, in effect, shrinking matrix A by the corresponding row and column, which of course affects **G**, reduces λ_{max}, and increases R.

The following discussion examines point, line and finally block iterative methods. The derivations, proofs and formulae associated with the selected iterative techniques are well established and appear elsewhere as noted throughout the discussion, therefore, we will not attempt to derive nor show the derivations of these methods. The discussion of each method will contain, where applicable and as required

- a brief description, including the handling of boundaries,
- a pictorial representation,
- an iteration formula,
- a formula for the spectral radius λ_{max}, and
- a formula for the optimal acceleration factor ω_{opt},

for the model problem in which we have a square with width normalized to π (unless noted otherwise), having uniform mesh spacing h, Dirichlet boundaries assumed on all four sides, and no singular points (ie. Dirichlet boundaries due to the corona) on the interior of the mesh. Since we *are* growing corona on the mesh, and we *may have* Neumann boundaries, the actual spectral radius and ω_{opt} will in fact different than the values obtained from the standard formulas. For the given square, we have $n = \pi/h$ mesh squares along any side, and a set of $(n-1)^2$ interior mesh points defined for

$$x = ih, \quad y = jh, \quad i, j = 1,2,\ldots,n-1, \quad \text{with} \quad nh = \pi. \tag{2.44}$$

For the formulas that follow below, i,j are the coordinates of an arbitrary point P on the mesh, $U_{i,j}$ is the potential at point $P(i,j)$, and U^k is the set of all mesh potentials at iteration step k.

2.4.4.2 Point Iterative Methods

Point iterative methods iterate all the points on the mesh, one point at a time. The general scanning convention is left to right, and top to bottom. If any singularities are encountered, they are skipped by the point methods.

2.4.4.2.1 Point Jacobi

The point Jacobi (relaxation) iterative technique is generally considered to be the reference point for performance comparisons of other iterative methods. The Jacobi method has the slowest convergence rate of all iterative methods, but it also happens to be the most stable numerical method, and can be used as a fallback method when other methods fail to converge for our model problem. The Jacobi method iterates the mesh points, point-by-point, simultaneously solving for all points on the mesh, using only the previous potentials (see Fig. 2.4). U^{k+1} is invariant of the order of scanning the mesh.

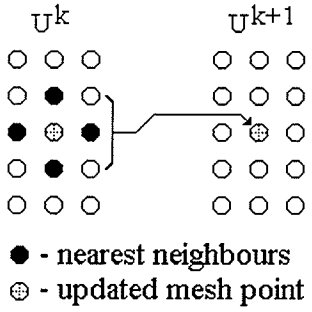


Fig. 2.4: Jacobi iteration process.

The point Jacobi iteration formula is given by

$$U^{k+1}_{i,j} = [\, U^{k}_{i,j-1} + U^{k}_{i+1,j} + U^{k}_{i,j+1} + U^{k}_{i-1,j} \,] / 4 \qquad (2.45)$$

The spectral radius for the Jacobi method applied to a square is given by

$$\lambda_{max} = \cos(h) \approx 1 - \frac{h^2}{2} \quad \text{as } h \to 0 \tag{2.46}$$

The spectral radius for the Jacobi method applied to a rectangular grid divided into $N \times M$ mesh squares, each being $h \times h$ in size (uniform mesh spacing h) and surrounded by Dirichlet boundaries is given by [PTVF92]

$$\lambda_{max} = \frac{\cos\left(\frac{\pi}{N}\right) + \cos\left(\frac{\pi}{M}\right)}{2} \quad \text{as } h \to 0 \tag{2.47}$$

Determining the spectral radius for other shapes is more complicated [Rand67, Lewi71], possibly requiring the analysis of an analogous membrane [LlMc68], or λ_{max} may be determinable only from some iteration norm [Reid66].

2.4.4.2.2 Point Gauss-Seidel

In Gauss-Seidel point iteration [Ames92, p. 200], all points are successively relaxed, using the improved values for adjacent points, as shown in Fig. 2.5. U^{k+1} is *not* scanning order invariant.

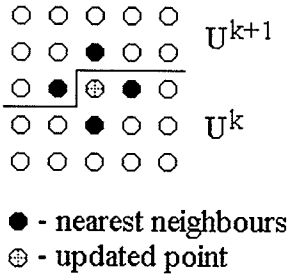


Fig. 2.5: Gauss-Seidel iteration process.

The point Gauss-Seidel iteration formula is given by

$$U^{k+1}_{i,j} = [\, U^{k+1}_{i,j-1} + U^{k}_{i+1,j} + U^{k}_{i,j+1} + U^{k+1}_{i-1,j} \,] / 4 \tag{2.48}$$

The spectral radius is given by

$$\lambda_{\max} = \cos^2(h) \approx 1 - h^2 \quad \text{as } h \to 0 \tag{2.49}$$

2.4.4.2.3 Point SOR

In successive over-relaxation (SOR) iteration, all points are *over*-relaxed *using* the improved values from the previously solved adjacent row. SOR involves an acceleration factor ω [Young54, Young71] to over-correct the potential at each point, hence the *over*-relaxation terminology.

The point SOR iteration formula is given by

$$U^{k+1}_{i,j} = U^{k}_{i,j} + \omega/4\,[\,U^{k+1}_{i,j-1} + U^{k}_{i+1,j} + U^{k}_{i,j+1} + U^{k+1}_{i-1,j} - 4U^{k}_{i,j}] \tag{2.50}$$

The spectral radius for a square at $\omega=\omega_{opt}$ is given by

$$\lambda_{\max} = \frac{1-\sqrt{1-\left(\lambda_{\max_{JACOBI}}\right)^2}}{1+\sqrt{1-\left(\lambda_{\max_{JACOBI}}\right)^2}} = \frac{1-\sin(h)}{1+\sin(h)} \approx 1-2h, \quad \text{as } h \to 0 \tag{2.51}$$

The acceleration factor ω_{opt} is given by

$$\omega_{opt} = 1 + \lambda_{\max} = \frac{2}{1+\sqrt{1-\left(\lambda_{\max_{JACOBI}}\right)^2}} = \frac{2}{1+\sin(h)}, \quad \text{as } h \to 0 \tag{2.52}$$

Some variations of the basic SOR method include red-black, conformal, and alternating direction ordering of the mesh scanning, and ad hoc [Ehrl81] and Chebychev [PTVF92] accelerations.

2.4.4.2.4 Successive Extrapolated Relaxation

Successive Extrapolated Relaxation (SER) [DeKi73a, DeKi73b] is a technique based on the extrapolated sum of a geometric series useful for extrapolating many,

possibly thousands, of iterations into the future. This method is currently not implemented due to deadline constraints, although it is intended to be added in the very near future.

The basis of this method is to store the complete mesh potential obtained for 3 successive global iterations performed at steps k, k+1, k+2. The previous 2 eigenvalue estimates for each mesh point can be computed. From this, a geometric series for the eigenvalues of each point is determined. Finally, the potentials at every mesh point can be extrapolated by extrapolating the sum of its geometric series and multiplying it by the point's potential. Several global iterations are then performed. The complete process is then repeated until some measure of convergence is attained.

An implementation based on extrapolated 2^{nd}-order curve fitting was attempted prior to obtaining the references above, and it was found that extrapolation beyond a few iterations yielded negative values for the potentials for most mesh points. This was not surprising since the monotonically decreasing potentials at each point where fitted with a parabolic function whose potential would inevitably cross the 0 potential and become negative as the iteration number increased. This led to the conclusion that the attempted implementation was wrong.

2.4.4.2.5 Other Point Iterative Methods

Other point iterative techniques exist, such as symmetric successive over-relaxation (SSOR) [DSMi63], which requires twice the effort of SOR for only half the convergence rate, and the extrapolated Aitken's method [Evan63, Evan64a]. Neither was implemented due to the availability of higher performance methods such as SOR.

2.4.4.3 Line Iterative Methods

Line methods iterate the mesh points, line-by-line, simultaneously solving for all points on a given line, while considering all adjacent points as frozen, analogous to point methods. In effect, all points along a line (row or column) are modified in a single step. The resulting set of equations for all points on the line of interest result in a tridiagonal matrix which is easily inverted [PTVF92]. If any singularities exist in the row being iterated, the row must be segmented into singularity-free segments which are then solved by line methods. Line iterative methods are sometimes referred to as block methods [Ames92].

2.4.4.3.1 Jacobi Line Over-relaxation

In Jacobi line over-relaxation (JLOR) [Ames77, p. 199], all points on successive lines are solved simultaneously *without* using the newly obtained values from the previously solved adjacent row (see Fig. 2.6), similar in essence to the point Jacobi method. This method is also known as the Jacobi row iterative method.

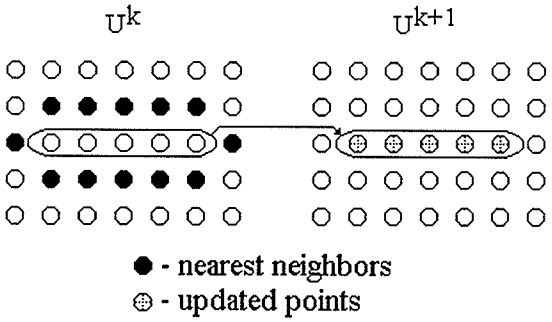


Fig. 2.6: JLOR performed on a single mesh line.

The iteration formula for the Jacobi row iteration method is

$$U^{k+1}_{i,j} = 1/4\,(U^{k+1}_{i+1,j} + U^{k+1}_{i-1,j} + U^{k}_{i,j+1} + U^{k}_{i,j-1}) \qquad (2.53)$$

and the spectral radius is given by

$$\lambda_{max} = \frac{\cos(h)}{2-\cos(h)} \approx 1-h^2, \quad \text{as } h \to 0 \tag{2.54}$$

This method was not implemented due to the extra storage requirements of this non-in-place method, and due to the consideration of the easily obtainable $2 \cdot 2^{1/2}$ increase in convergence available in SLOR.

2.4.4.3.2 Gauss-Seidel Row Iteration

In Gauss-Seidel row iteration [Ames92, p. 200], all points on successive lines are solved simultaneously using the newly obtained values from the previously solved adjacent row, similar in essence to the point Gauss-Seidel, as shown in Fig. 2.7.

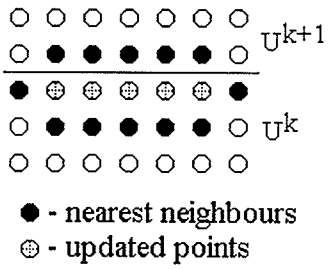


Fig. 2.7: Gauss-Seidel row iteration process.

The iteration formula for the Gauss-Seidel row iteration method is

$$U^{k+1}_{i,j} = 1/4\,(U^{k+1}_{i+1,j} + U^{k+1}_{i-1,j} + U^{k}_{i,j+1} + U^{k+1}_{i,j-1}) \tag{2.55}$$

and the spectral radius is given by

$$\lambda_{max} \approx 1-2h^2, \quad \text{as } h \to 0 \tag{2.56}$$

This method was implemented for comparison with SLOR.

2.4.4.3.3 Successive Line Over-Relaxation

In successive line over-relaxation (SLOR) iteration [Ames77, p. 199, Evan62], all points on successive lines are over-relaxed simultaneously *using* the newly obtained

values from the previously solved adjacent row, and involving an acceleration factor ω, similar in essence to the point SOR.

The iteration formula for the SLOR method is

$$U^{k+1}_{i,j} = U^{k}_{i,j} + \omega/4\,(U^{k+1}_{i+1,j} + U^{k+1}_{i-1,j} + U^{k}_{i,j+1} + U^{k+1}_{i,j-1} - 4U^{k}_{i,j}) \tag{2.57}$$

The optimal acceleration factor ω_{opt} and spectral radius λ_{max} are given by

$$\lambda_{\max_{(SLOR)}} = \omega_{opt} - 1 \approx 1 - 2\sqrt{2}h, \quad \text{as } h \to 0 \tag{2.58}$$

$$\omega_{opt} = 1 + \frac{1 - \sqrt{1 - \left(\lambda_{\max_{JLOR}}\right)^2}}{1 + \sqrt{1 - \left(\lambda_{\max_{JLOR}}\right)^2}} \approx 2 - 2\sqrt{2}h, \quad \text{as } h \to 0 \tag{2.59a}$$

An alternative formula for ω_{opt} for a square of area A which is dependent on the mesh size h as h approaches 0, is derived by Evans [Evan62] as

$$\omega_{opt} = \frac{4}{3\left(1 + \frac{2.405\sqrt{\pi}}{3\sqrt{A}}h\right)} \tag{2.59b}$$

where

$$2.405\sqrt{\frac{\pi}{A}} \le \lambda_{Jac}$$

is a lower bound of the corresponding Jacobi spectral radius λ_{Jac} [FoWa67, p. 265].

2.4.4.3.4 Successive 2-Line Over-Relaxation

Successive 2-line over-relaxation (S2LOR) is similar to SLOR with the exception that 2 adjacent rows are solved simultaneously. The resulting equations produce a matrix which is no longer tridiagonal, but block-tridiagonal, requiring more time and effort to

solve. S2LOR converges faster than SLOR by a factor of $2^{1/2}$ for "nice", singular-free problems [Ames77, p. 201].

The main prohibition against applying S2LOR to our problem is the severe segmentation caused by non-aligned, scattered singularities which may occur throughout the 2 lines to be solved. The algorithm would need to segment the 2 lines into smaller,

1. singularity-free 2-line segments to be solved by SLOR and possibly altering the required acceleration factor, and
2. 2-line segments containing at least 1 singularity in each column, to be solved by SOR.

Figure 2.8 shows the required segmentation for 4 lines of a typical 14×14 mesh containing a corona discharge.

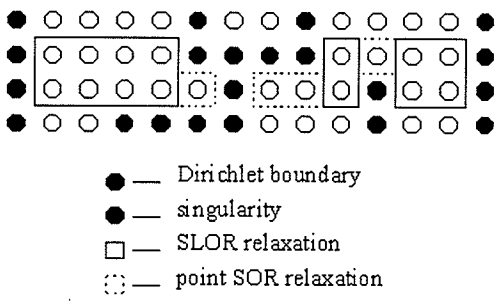


Fig. 2.8: S2LOR segmentation example.

This segmentation would slow down the algorithm to the point of possibly losing the ideal $2^{1/2}$ increase in the convergence rate.

2.4.4.3.5 Alternating Direction, Successive Line Over-Relaxation

Alternating direction, successive line over-relaxation (ADSLOR) [Evan64b] is a two-pass method in which SLOR is performed first on all the rows, and then on all the columns of the mesh.

The iteration formula for the row-wise solution is

$$U^{k+1/2}_{i,j} = U^{k}_{i,j} + \omega/4\,(U^{k+1/2}_{i+1,j} + U^{k+1/2}_{i-1,j} + U^{k}_{i,j+1} + U^{k+1/2}_{i,j-1} - 4U^{k}_{i,j}) \qquad (2.60)$$

The iteration formula for the column-wise solution is

$$U^{k+1}_{i,j} = U^{k+1/2}_{i,j} + \omega/4\,(U^{k+1/2}_{i+1,j} + U^{k+1}_{i-1,j} + U^{k+1}_{i,j+1} + U^{k+1}_{i,j-1} - 4U^{k+1/2}_{i,j}) \qquad (2.61)$$

A formula for ω_{opt} for a *unit* square of area A which is dependent on the mesh spacing h as h approaches 0 is [Evan64b]

$$\omega_{opt} = \frac{4}{3\left(1 + \frac{2.405\sqrt{\pi}}{3\sqrt{2}\sqrt{A}} \cdot h\right)}, \quad \text{as } h \to 0 \qquad (2.62)$$

2.4.4.3.6 Alternating Direction Implicit

Many alternating direction implicit (ADI) methods have been developed, the foremost of which are the Peaceman-Rachford ADI or PRADI (1955), Douglas-Rachford (1956) ADI or DRADI, the Douglas (1962) implementations [Hadj68], and others [FaMi66]. ADI is essentially a two-pass method; in the first pass all the rows are sequentially solved, followed by a second pass in which all the columns are sequentially solved. These methods speed up convergence by the use of a precalculated sequence of acceleration factors, or by the use of multiple optimal parameters which may depend on the iteration step! The calculation of the parameters or acceleration factors is very involved [Ames77, p. 149]. ADI iterative techniques have also been extended to non-rectangular [Lewi71] and to 3-dimensional problems [Hadj68, Hadj71].

Due to the shrinking of **A** (Sec. 2.4.4.1), there is no guarantee that sequence of acceleration factors or the optimal parameters are invariant. Since there are

computationally faster and simpler techniques available, ADI was not implemented nor evaluated.

2.4.4.4 Block Iterative Methods

Block iterative techniques iterate the mesh points, block-by-block, simultaneously solving for all points within a given block, while considering all adjacent points as constant. In effect, all points within a block are modified in a single step [ShWe38]. The resulting set of equations for all points within the block result in a full matrix which is not easily inverted.

Blocks cannot be segmented in order to handle enclosed singularities, therefore to be considered as viable solution techniques, block iterative methods must be capable of handling any number of singularities enclosed within any given block.

2.4.4.4.1 Successive Block Over-Relaxation

In successive block over-relaxation (SBOR) iteration [Evan84], all points in an m×m block are over-relaxed simultaneously using the newly obtained values from the previously solved adjacent blocks (see Fig. 2.9) and involving an acceleration factor ω, similar in essence to the point and line SOR.

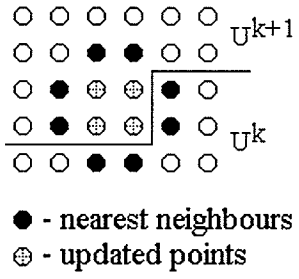


Fig. 2.9: Example of 2×2 SBOR.

An estimate of the spectral radius of k×k SBOR on a *unit square* [MaTe61], and independent of k (apparently, unless they used k=2) is given as

$$\lambda_{max} = 1 - 2\sqrt{2}\pi h, \quad \text{as } h \to 0 \tag{2.63}$$

which transformed for the reference π×π square is

$$\lambda_{max} = 1 - 2\sqrt{2} h, \quad \text{as } h \to 0 \tag{2.64}$$

Another formula, which depends on k and is derived from [Part81] is proposed as

$$\lambda_{max} = 1 - 2\sqrt{2}\sqrt{\frac{k}{2}} h, \quad \text{as } h \to 0 \tag{2.65}$$

A problem arises when we extend the method to handle:

1. different boundary situations: all Dirichlet boundaries, or Neumann boundaries on the left or right columns of the 2×2 block, and
2. any combination ($2^4 = 16$ unique combinations) of singularities enclosed by the block.

Assuming that we are using pre-calculated, *non-rotatable* stencils to speed up computations, the above yields a total of 3×16 = 48 permutations of possible boundaries, each of which requires its own unique stencil. The complete derivation of the permutations of the 2×2 block stencil appears in Appendix C.

For 3×3 blocks, we again have 3 different boundary situations, but we have 2^9 combinations of enclosed singularities, for a total of $3 \times 2^9 = 1536$ permutations, each requiring a unique stencil. Clearly, the inclusion of singularities has undesirable effects in the implementation of 3×3 (and larger) block methods. The author is aware of SBOR being employed with only 2×2 and 3×3 blocks [Evan84, ShWe38]. A more detailed discussion and derivation appears in Appendix C.

2.4.4.4.2 Alternating Direction, Successive Block Over-Relaxation

Alternating direction, successive block over-relaxation (ADSBOR) is the author's extension of SBOR, and as such, there are no known formula's for the spectral radius or optimal acceleration factor. ADSBOR is a two-pass method in which SBOR is performed first row-wise, and then column-wise on the mesh points, similar to ADSLOR.

2.4.4.5 K-Line and K×K Block Methods

K-*line methods*, also referred to as block methods by some [CuVa59, Fabe81], simultaneously operate on *k-lines of the matrix* **A**. This corresponds to *k mesh points*, not necessarily one complete mesh row, and possibly split into two segments by a mesh edge. K×K *block methods* attempt to simultaneously operate on a *k×k block of the matrix* **A** [CuVa59, Fabe81], or they attempt to simultaneously solve a block of *k×k mesh points* by block relaxation [Part81].

Since our model problem uses a large mesh and direct methods have been ruled out, there is no explicit matrix, so neither k-line nor k×k block (Fabe81 definition) iteration methods are explicitly implemented. It should be noted, however, that whenever the implemented line methods encounter singularities and segmentation is performed to obtain singularity-free segments, we are in effect reverting to k-line methods.

2.4.4.6 Convergence Rates

Table 2.1: Convergence Rates for Iterative Methods (5-Point Stencil).

Iteration Method	Convergence Rate		Relevant Section
	$\pi\times\pi$ square	1×1 square	
point Jacobi	$\frac{h^2}{2}$	$\frac{\pi^2 h^2}{2}$	2.4.4.2.1
point Gauss-Seidel	h^2	$\pi^2 h^2$	2.4.4.2.2
point SOR	$2h$	$2\pi h$	2.4.4.2.3
Jacobi line relaxation	h^2	$\pi^2 h^2$	2.4.4.3.1
Gauss-Seidel line relaxation	$2h^2$	$2\pi^2 h^2$	2.4.4.3.2
SLOR	$2\sqrt{2}h$	$2\sqrt{2}\pi h$	2.4.4.3.3
S2LOR	$4h$	$4\pi h$	2.4.4.3.4
ADSLOR	$\geq 2\sqrt{2}h$	$\geq 2\sqrt{2}\pi h$	2.4.4.3.6
SBOR (2×2)	h^2	$\pi^2 h^2$	2.4.4.4.1
SBOR (3×3)	$\sqrt{3/2}h^2$		2.4.4.4.1
ADSBOR (2×2)	$\geq h^2$		2.4.4.4.2
ADSBOR (3×3)	$\geq \sqrt{3/2}h^2$		2.4.4.4.2

2.4.4.7 Convergence

An important issue that all iteration schemes must address is assurance of convergence, and the stopping criteria employed to halt the iterations. What is required is a measure of the impact of the iteration on the convergence process, and a method of relating it to some convergence criteria.

2.4.4.7.1 Assurance of Convergence

The question that might be posed is whether the selected iterative method will cause $\mathbf{u}^k$ to converge to the exact solution $\underline{\mathbf{u}}$. Young [Young54, YoGr73] defined the

conditions and characteristics of a certain class of matricies possessing what he defined as Property (A). The conditions for a matrix to possess Property (A), by rearrangement if necessary, are:

- all coefficients are real [Ames77, p. 112],
- positive diagonal ($a_{ii} > 0$) and , $a_{ij} \leq 0$ for $i \neq j$ [Ames77, p. 101],
- diagonal dominance [Ames77, p. 101],
- symmetric, so that $\mathbf{A} = \mathbf{A}^T$, which also implies $\mathbf{A}$ is tridiagonal or block tridiagonal,

For any system $\mathbf{Au} = \mathbf{b}$ derived from the 5-point operator, matrix $\mathbf{A}$ possesses property (A), therefore:

- its eigenvalues are linearly independent and positive [Ames77, p 107] since $\mathbf{A}$ is symmetric,
- ω_{opt} is independent of scanning order, and dependent only on λ_{max} [FoWa67, p. 261],
- the point Jacobi and Gauss-Seidel methods always converge, and the SOR method always converges for $0 < \omega < 2$. [Ames77, p. 113].

In fact, since $\mathbf{A}$ is positive definite, then *any* point, line, or block *iterative method* which reduces the residual $\mathbf{r}^k$ to 0 by modifying U sufficiently often will lead to convergence [MaTe61]. It is known that the 9-point difference schemes yield matrices which are block tridiagonal but which do not possess Property (A) [Evan64b]. This does, however, not preclude their convergence.

2.4.4.7.2 Residuals

In our model problem, we are trying to solve $\mathbf{Au} = \mathbf{b}$ by some iterative technique such that

$$\mathbf{u}^{k+1} = \mathbf{G}\mathbf{u}^{k} + \mathbf{v} \tag{2.66}$$

where $\mathbf{G}$ is the iteration matrix, dependent on the iteration method. If $\underline{\mathbf{u}}$ is the exact solution that we are after, where $\underline{\mathbf{u}} = \mathbf{A}^{-1}\mathbf{b}$, then the absolute error vector $\mathbf{e}^k$ can be expressed as

$$\mathbf{e}^k = |\mathbf{u}^k - \underline{\mathbf{u}}| = |\mathbf{u}^k - \mathbf{A}^{-1}\mathbf{b}| \tag{2.67}$$

and we can declare convergence when $\mathbf{e}^k = 0$. Since, however, $\underline{\mathbf{u}}$ is unknown and $\mathbf{A}^{-1}\mathbf{b}$ is intractable, we must resort to other methods.

One approach is to use residuals [MaTe61, FoWa67, Ames77, Smit85], in which the *residual* vector $\mathbf{r}$ at iteration step k is defined as

$$\mathbf{r}^k = \mathbf{A}\mathbf{u}^k - \mathbf{b} \tag{2.68}$$

The residual at any non-singular mesh point P(i,j) is the amount by which the finite difference equation Eq. 2.19 differs from zero. The calculation of $\mathbf{r}^k$ must be done after all points have been updated at any iteration step k, requiring that the entire mesh be rescanned, regardless of the iteration method used. Caution must be exercised since the absolute error (Eq. 2.67) may be large [Smit65, p. 158], even though the residuals are small.

The pseudo-residual [YoKi81] or *displacement vector* $\boldsymbol{\delta}$, [Smit65, p. 78] is the amount by which $\mathbf{u}$ has been changed in one iteration, and is defined as

$$\boldsymbol{\delta}^k = \mathbf{u}^k - \mathbf{u}^{k-1} \tag{2.69}$$

$\boldsymbol{\delta}^k$ is easily calculated *during* the iteration sweep for all *point* methods, however, this method is not easily extendible to non-Jacobi line and block methods as they simultaneously update multiple mesh points *in-place* for any iteration step k, over-writing the previous values without preserving them. The overhead penalty for line methods to

preserve in a buffer, N values at a time, for each of the M rows, for each iteration sweep would be considerable.

Rewriting for comparison the previous two equations for any point P(*i,j*) on our mesh, the residual is defined as

$$r^k_{i,j} = [\, U^k_{i,j-1} + U^k_{i+1,j} + U^k_{i,j+1} + U^k_{i-1,j} - 4U^k_{i,j}] \tag{2.70}$$

and the displacement vector for point SOR can be written as

$$\delta^k_{i,j} = U^k_{i,j} - U^{k-1}_{i,j} = \omega/4\, [\, U^k_{i,j-1} + U^{k-1}_{i+1,j} + U^{k-1}_{i,j+1} + U^k_{i-1,j} - 4U^{k-1}_{i,j}] \tag{2.71}$$

2.4.4.7.3 Norms

For a vector **V** of size *n*, with *i*=0,1,...,*n*-1, the first (-power) norm is defined as [FoWa67, p. 370]

$$\|V\|_1 = \sum_i |V_i| \tag{2.72}$$

the second, or Euclidean norm [FoWa67, p. 205] is defined as

$$\|V\|_2 = \left(\sum_i V_i^2 \right)^{1/2} \tag{2.73}$$

and the infinite, maximum, or Chebychev norm [FoWa67, p. 321] is defined as

$$\|V\|_\infty = \max \left(|V_i| \right) \tag{2.74}$$

The second norm is the most difficult norm to compute, suffering the greatest loss of accuracy, the maximum norm is the easiest to compute and suffers no loss of accuracy, and the first norm lies somewhere between [FoWa67, p 258].

As a direct consequence result of the properties and the equivalent eigenvector composition of the iteration matrix **G**, we can *estimate* the spectral radius of the iteration

methods from the ratios of successive residual or displacement norms. Expressed in mathematical terms, the *estimate* of λ_{max} is

$$\frac{\left\|\mathrm{r}^{k}\right\|_{p}}{\left\|\mathrm{r}^{k-1}\right\|_{p}} = \hat{\lambda}_{\max}, \quad \text{for } p = 1, 2, \infty \tag{2.75}$$

which for any norm p of the *residual* vector [FoWa67, p. 257] converges to λ_{max},

$$\lim_{k\to\infty} \frac{\left\|\mathrm{r}^{k}\right\|_{p}}{\left\|\mathrm{r}^{k-1}\right\|_{p}} = \lambda_{\max}, \quad \text{for } p = 1, 2, \infty \tag{2.76}$$

and similarly, for any norm p of the *displacement* vector [FoWa67, p. 369]

$$\lim_{k\to\infty} \frac{\left\|\delta^{k}\right\|_{p}}{\left\|\delta^{k-1}\right\|_{p}} = \lambda_{\max}, \quad \text{for } p = 1, 2, \infty \tag{2.77}$$

This yields 6 different approaches to estimating λ_{max}, but as of yet, we do not know which approach will converge to λ_{max} the fastest or the smoothest. The method best suited for our problem remains unknown at this point.

2.4.4.7.4 Convergence Criteria

There appears to be a great lack of information regarding the stopping procedures used in published experiments. Some [Aren94] halt the iterations when the worst case error (implies residual method), is less than some threshold ξ so that

$$\left\|\mathrm{r}^{k}\right\|_{\infty} = \max\left(\left|\mathrm{r}_{i,j}{}^{k}\right|\right) \leq \xi \tag{2.78}$$

Others [Evan63] halt when a norm has been reduced by some factor, such as

$$\frac{\left\|\mathrm{r}^{k}\right\|}{\left\|\mathrm{r}^{0}\right\|} \leq \xi \tag{2.79}$$

or when the maximum displacement is less than a threshold [FaMi66] such that

$$\left\| \delta^k \right\|_\infty \leq \xi \tag{2.80}$$

2.4.4.7.5 Acceleration of Convergence

For accelerated iterative methods, acceleration factor ω and the spectral radius λ_{max} are inter-related. At some optimal value for ω called ω_{opt}, λ_{max} is minimized, maximizing the convergence rate R. For problems with simple, generally symmetrical shaped, singular-free problems, ω_{opt} can be determined prior to any iterations. For problems with complex shapes, or containing singularities, determination of ω_{opt} is intractable or impossible. For these situations, which include our model problem, the spectral radius of G must be estimated by the methods described in Sec. 2.4.4.7.3, in order to compute an *estimate* of the optimum acceleration parameter ω_{opt} [Carr61, Reid66, FoWa67, Erhl81]. An estimate $\hat{\omega}_{opt}$ of ω_{opt} is far better than using ω=1 since

$$R\left(\omega_{opt}\right) \geq R\left(\hat{\omega}_{opt}\right) >> R(1.0) \tag{2.81}$$

2.5. Fractal Dimensions

2.5.1 Introduction

After modelling and generating dielectric breakdown discharges using proper modelling and appropriate computational methods, it is desirable to qualitatively compare the different structures obtained from a set of simulations in which certain parameters are varied. From an observer's view, the resultant structures will have similar characteristics and will *appear to be similar* even though the *shapes will be different*, and it is our desire to compare these structures. From previous research, it is known that these structures

possess fractal and multifractal properties [HaMP86, PiEW86, WiPi86, Ever90, Vics92, KiSt94, Aren94]. These structures are therefore called *Laplacian multifractals*, or *Laplacian fractals* [WiPi86] for short, since their growth process is governed by the solution of Laplace's equation for the potential field, and their structure exhibits global scaling properties best described by fractal and multifractal measures.

There are several defining characteristics associated with fractal objects. Fractal objects generally:

- are the resultant structures from some process [PeJS92, p. viii], often a chaotic one,
- have self-similar properties whose measurement exhibit global scaling properties, for which the fractal dimension may be (simply) defined as the critical exponent which makes that measure constant over all scales,
- are often non-differentiable, or even nowhere differentiable, such as the Takagi curve [Kins94f , p. IV-37], and
- are continuous, discontinuous, or discontinuous everywhere such as the Cantor dust.

2.5.2 Definition of Fractal Dimensions

Mandelbrot defined a fractal object [Kins94e] as

"a fractal is the set for which the Hausdorff-Besicovitch dimension strictly exceeds the topological (Euclidean) dimension."

The Hausdorff-Besicovitch dimension provides the basis for all morphological dimensions, and provides a fundamental role in the concepts of fractals. If we consider an n-dimensional Euclidean space $\Re^n$, where F (the fractal of interest) is a subset of $\Re^n$, s and δ are positive real numbers, $\delta > 0$, and V_j is a δ-cover of F, then the Hausdorff s-

dimensional measure of F is defined as [Kins94e, Stac95], then

$$H_{\delta}^{s} = \inf\left\{\sum_{j=1}^{\infty}\left\|V_j\right\|_2^s \,\middle|\, \{V_j\} \in F\right\} \tag{2.82}$$

Vj are vels (volume elements) of radius

$$\left\|V_j\right\|_2 \leq \delta \tag{2.83}$$

which are used to completely cover the fractal F so that

$$F = \bigcup_{j=1}^{\infty} V_j \tag{2.84}$$

The interpretation of this is that the covering sets V_j form the minimal δ-cover of F with a maximum volume element (vel) size not exceeding δ [Kins94e, p. 4]. As δ approaches 0, the Hausdorff measure increases. The Hausdorff measure becomes infinite if s is too small, or vanishes if s is too large. At some critical value D_{HB} of s, called the Hausdorff-Besicovitch dimension, the measure of F becomes constant, and is described by

$$D_{HB} = \inf\left\{s \,\middle|\, H_{\delta\to 0}^{s}(F) = 0\right\} = \sup\left\{s \,\middle|\, H_{\delta\to 0}^{s}(F) = \infty\right\} \tag{2.85}$$

The significance of D_{HB} is illustrated in Fig. 2.10. In practice, the Hausdorff measure is impractical because it is very difficult or impossible to compute. The mesh (or box) counting dimension is one of several practical implementations which estimates the Hausdorff dimension [PeJS92, p. 218].

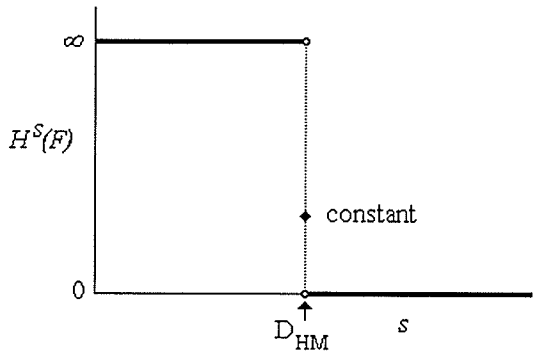


Fig. 2.10: Hausdorff-Besicovitch measure for some fractal *F*.

2.5.3 Classifications of Fractal Dimensions

The taxonomy of fractal measures includes several classifications of fractal measures [Kins94f, p. II-124] (or dimensions), based on the properties that they measure. *Morphological*-based dimensions measure the scaling properties of the geometrical complexity or coverage of the structure, but ignore all time-dependent behaviour of the object, and the distribution of the measure over the fractal. *Entropy*-based dimensions take into account the distribution of the measure being employed. They can measure the amount of information needed to specify the state of a fractal to an accuracy *r*, and as such can be used to measure the compressibility for data storage. They can also be used to measure the density distribution of spatial fractals to some resolution *r*. *Spectral* dimensions measure the scaling properties of the power spectrum, such as the persistence, generally of some temporal signal or image, and permit batch processing only. *Variance*-based dimensions permit a temporal or spatial signal to be analysed directly in real-time [Kins94d] by measuring the scaling properties of the signal's variance (σ).

2.5.4 Hausdorff Mesh Counting Dimension

Perhaps the best known of all dimension measures is the morphological-based box or (Hausdorff) mesh counting dimension D_{HM}, also referred by some as the capacity measure of the fractal [Grass83a]. Simply put, it relates the number of non-overlapping square vels N_r of width r needed to completely cover the fractal (normalized to a unit square) when it is being measured at a scale of $1/r$. The relationship can be expressed by the power law

$$N_r = \left(\frac{1}{r}\right)^{D_{HM}} \tag{2.86}$$

Solving for D_{HM}, we obtain in the limit

$$D_{HM} = \lim_{r \to 0} \frac{\log(N_r)}{\log(1/r)} \tag{2.87}$$

In practice, however, r is not infinitesimally small and $r_{min} >> 0$ (r_{min} = 0.002 for a 500×500 grid) so we must estimate D_{HM} by the slope of the line that best fits the $\log(N_r)$ and $\log(1/r)$ data obtained for a succession of vels with different sizes r_k [Kins94c, p. 3], an approach which is generally used for all fractal dimensions. Therefore, for the k^{th} stage of computation, N_k vels of size r_k are intersected by the fractal F and Eq. 2.87 can be expressed as

$$D_{HM} = \lim_{k \to \infty} \frac{\log(N_k)}{\log(1/r_k)} \tag{2.88}$$

Figure 2.11 illustrates the mesh counting dimension computed on the results for two scales. The number of mesh squares required to cover the fractal at each scale are counted and tabulated. Finally the fractal dimension of the object is the slope of the line that best

fits the log(N) and log (1/s) data. In actual implementations, the fractal is measured at more than just 2 scales.

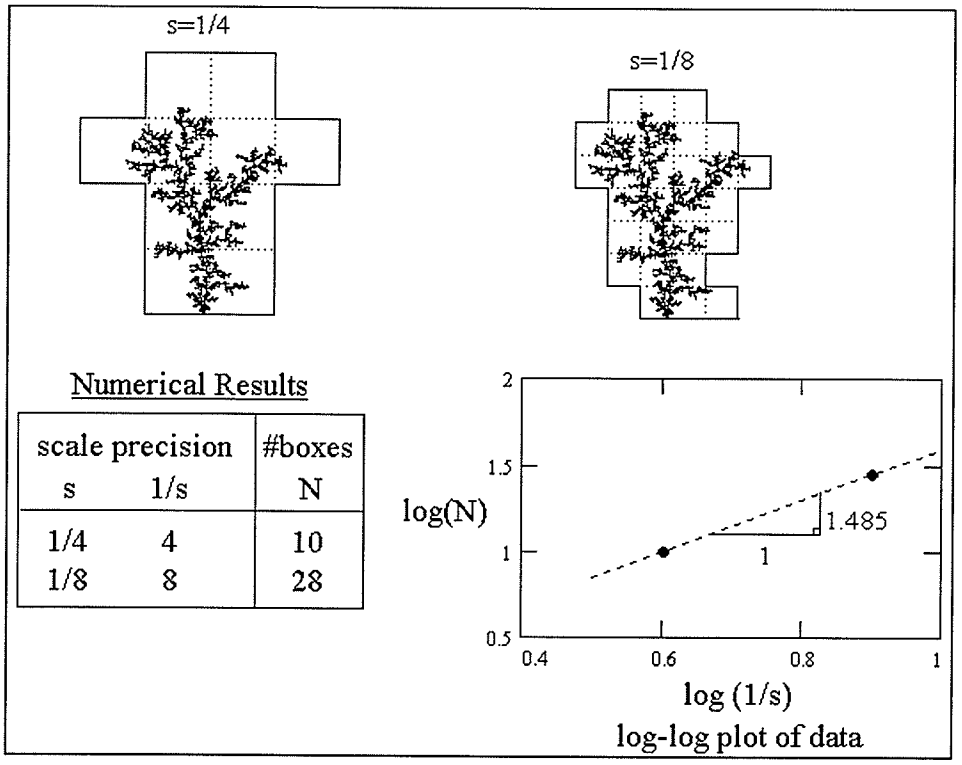


Fig. 2.11: Illustration of the mesh counting dimension.

The Hausdorff dimension is used to calculate the similarity dimension D_S of non-self-similar fractals such as our Laplacian fractals [Grass83a, Kins94f, p. II-38]. The Hausdorff mesh counting dimension D_{HM} will at worst over-estimate D_{HB} [Kins94f, p. II-119]. It should be noted that the mesh counting dimension suffers from slow convergence, and brute force implementations require large amounts of memory [Grass83a].

2.5.5 Information Dimension

An improvement on the mesh counting dimension is the information dimension D_I which measures how the entropy (distribution or disorder of probabilities) scales with

varying spatial resolution $1/r$ [AtSV88]. Unlike the mesh counting dimension, D_I takes into account that certain regions (mesh squares) intersect the fractal F more often than other regions. If we cover a fractal consisting of N points with vels of size (scale) $1/r$, the probability p_j that the trajectory of the fractal intersects the j^{th} vel, or that points of the fractal are found in the j^{th} vel, is given by

$$p_j = \frac{n_j}{N} \tag{2.89}$$

where n_j is the number of points found in that vel. The entropy H_r of the fractal is

$$H_r = -\sum_{j=1}^{N} p_j \log p_j \tag{2.90}$$

H_r can be interpreted as the minimum information required to specify a point on the fractal with accuracy r [GrPr83a] or the amount of information needed to specify the state of the fractal to an accuracy of r [Kins94f]. If we assume a power-law relationship between H_r and r, then the information dimension can be described by

$$D_I = \lim_{k\to\infty} \frac{H_r}{\log\left(1/r_k\right)} = \lim_{k\to\infty} \frac{\sum_{j=1}^{N_k} p_j \log p_j}{\log r_k} \tag{2.91}$$

2.5.6 Generalized Entropy Dimensions

Rényi developed the concept of generalized entropy from which we can construct a continuous spectrum of an infinite number of dimensions. It can be used to extract dimensionalized features from objects being measured [FeKi95], and it is the basis for the construction of the Mandelbrot dimension (Sec. 2.5.8).

2.5.6.1 Derivation

The generalized entropy, raised to any exponent q is expressed as [GrPr83b, HePr83, Kins94f, Chen95]

$$H_q = \frac{1}{q-1}\log\sum_{j=1}^{N} p_j^q, \quad -\infty \le q \le +\infty \tag{2.92}$$

and the generalized entropy (or Rényi) dimension is

$$D_q = \lim_{r\to 0}\frac{1}{q-1}\frac{\log\sum_{j=1}^{N_r} p_j^q}{\log(r)} \tag{2.93}$$

or, for the k^{th} stage of computation

$$D_q = \lim_{k\to\infty}\frac{1}{q-1}\frac{\log\sum_{j=1}^{N_k} p_j^q}{\log(r_k)} \tag{2.94}$$

For computational reasons, we can eliminate the division required by p_j (Eq. 2.89), if we express Eq. 2.94 as

$$D_q = \lim_{k\to\infty}\frac{1}{q-1}\frac{\log\left(\sum_{j=1}^{N_k} n_j^q\right) - q\log N_k}{\log(r_k)} \tag{2.95}$$

The dimension estimate can only *approach* the real value because N and r are finite. In the Laplacian fractals that we desire to measure, N is only of the order of 16,000 to 25,000, whereas if we would desire to measure a strange attractor, 10^6 points could easily be generated and analysed. Therefore, as in the previous dimensions, the dimension D_q is the slope of the best fitting line for the calculated H_k and r_k values.

2.5.6.2 Relation To Morphological Dimensions

Of the infinite fractal spectrum D_q, only D_0, D_1, and D_2 can be related to the morphology of the fractal F.

If $q = 0$, then Eq. 2.94 becomes

$$D_0 = \lim_{k\to\infty} \frac{1}{0-1} \frac{\log \sum_{j=1}^{N_k} p_j^0}{\log(r_k)} = \lim_{k\to\infty} \frac{\log \sum_{j=1}^{N_k} 1}{-\log(r_k)} = \lim_{k\to\infty} \frac{\log(N_k)}{\log(1/r_k)} \tag{2.96}$$

which shows that D_0 is equivalent to the Hausdorff mesh counting dimension D_{HM}, and is therefore also equivalent to the similarity dimension D_S. This can be expressed as

$$D_{HM} = D_S = D_0 \tag{2.97}$$

If $q = 1$, then Eq. 2.94 will be singular, so if we change the form of p_j [GrPr83b, Kins94f], use natural logarithms in Eq. 2.94 and use the series expansion for e^x, then

$$p_j^q = p_j p_j^{q-1} = p_j e^{(q-1)\ln p_j} \approx p_j\left(1+(q-1)\ln p_j\right) \tag{2.98}$$

Substituting Eq. 2.98 into Eq. 2.94 we have

$$D_1 \approx \lim_{k\to\infty} \frac{1}{q-1} \frac{\ln \sum_{j=1}^{N_k} p_j\left(1+(q-1)\ln p_j\right)}{\ln(r_k)} = \lim_{k\to\infty} \frac{1}{q-1} \frac{\ln\left(1+(q-1)\sum_{j=1}^{N_k} p_j \ln p_j\right)}{\ln(r_k)} \tag{2.99}$$

Using the series expansion for $\ln(1 + x)$, then reverting to base 10 logarithms, we arrive at

$$D_1 = \lim_{k\to\infty} \frac{1}{q-1} \frac{(q-1)\sum_{j=1}^{N_k} p_j \ln p_j}{\ln(r_k)} = \lim_{k\to\infty} \frac{1}{q-1} \frac{(q-1)\sum_{j=1}^{N_k} p_j \log p_j}{\log(r_k)} = \lim_{k\to\infty} \frac{H_k}{\log(1/r_k)} \tag{2.100}$$

Comparing Eqs. 2.91 and 2.100, we observe that D_1 is equivalent to the information dimension D_I.

If $q = 2$, then Eq. 2.94 becomes

$$D_2 = \lim_{k\to\infty} \frac{1}{2-1} \frac{\log \sum_{j=1}^{N_k} p_j^2}{\log(r_k)} = \lim_{k\to\infty} \frac{\log \sum_{j=1}^{N_k} p_j^2}{\log(r_k)} \tag{2.101}$$

which shows that D_2 is equivalent to the pair-correlation dimension D_C, derived elsewhere [Kins94f]. For $q > 2$, D_q can be interpreted as the q-tuple correlation dimensions of the fractal F.

2.5.6.3 Extrema of D_q

Since D_q is a spectrum of an infinite number of dimensions, it is intractable to calculate them all. In fact, for our Laplacian fractals, we find sufficient detail extracted for the range -60 ≤ q ≤ 60. We are however, also interested in the extrema of D_q, namely $D_{-\infty}$ and D_{∞}.

For $D_{-\infty}$, if we multiply the summation inside the log in Eq. 2.94 by $(p_{min}{}^q/p_{min}{}^q)$, where p_{min} is the minimum probability, we obtain

$$D_{-\infty} = \lim_{k\to\infty} \lim_{q\to-\infty} \frac{1}{q-1} \frac{\log\left(\sum_{j=1}^{N_k} \left(\frac{p_{\min}^q}{p_{\min}^q} p_j^q\right)\right)}{\log(r_k)} = \lim_{k\to\infty} \lim_{q\to-\infty} \frac{1}{q-1} \frac{\log\left(p_{\min}^q \sum_{j=1}^{N_k} \left(\frac{p_j}{p_{\min}}\right)^q\right)}{\log(r_k)}$$

$$D_{-\infty} = \lim_{k\to\infty} \lim_{q\to-\infty} \frac{1}{q-1} \frac{\log\left(p_{\min}^q\right)}{\log(r_k)} = \lim_{k\to\infty} \lim_{q\to-\infty} \frac{q}{q-1} \frac{\log(p_{\min})}{\log(r_k)} = \lim_{k\to\infty} \frac{\log(p_{\min})}{\log(r_k)} \tag{2.102}$$

For $D_{+\infty}$, if we multiply the summation inside the *log()* in Eq. 2.94 by $(p_{max}{}^q/p_{max}{}^q)$, where p_{max} is the maximum probability, we obtain

$$D_{+\infty} = \lim_{k\to\infty}\lim_{q\to\infty}\frac{1}{q-1}\frac{\log\left(\sum_{j=1}^{N_k}\left(\frac{p_{\max}^q}{p_{\max}^q}p_j^q\right)\right)}{\log(r_k)} = \lim_{k\to\infty}\lim_{q\to\infty}\frac{1}{q-1}\frac{\log\left(p_{\max}^q\sum_{j=1}^{N_k}\left(\frac{p_j}{p_{\max}}\right)^q\right)}{\log(r_k)}$$

$$D_{+\infty} = \lim_{k\to\infty}\lim_{q\to\infty}\frac{1}{q-1}\frac{\log\left(p_{\max}^q\right)}{\log(r_k)} = \lim_{k\to\infty}\lim_{q\to\infty}\frac{q}{q-1}\frac{\log(p_{\max})}{\log(r_k)} = \lim_{k\to\infty}\frac{\log(p_{\max})}{\log(r_k)} \qquad (2.103)$$

Thus we can see the sifting property associated with q in that it can suppress or emphasize the probabilities p_j.

2.5.6.4 Other Properties of D_q

If the fractal F has a *homogeneous probability distribution* such that $p_j = 1/N_k$, then

$$D_q = \lim_{k\to\infty}\frac{1}{q-1}\frac{\log\left(\sum_{j=1}^{N_k}\left(\frac{1}{N_k}\right)^q\right)}{\log(r_k)} = \lim_{k\to\infty}\frac{1}{q-1}\frac{\log\left(N_kN_k^{-q}\right)}{\log(r_k)} = \lim_{k\to\infty}\frac{\log(N_k)}{\log\left(1/r_k\right)} \qquad (2.104)$$

from which we conclude that $D_q = D_{HM} = D_S$ (from Section 5.4) for all q, implying that F is self-similar. Generally, most fractals are not self-similar, and the difference between D_q's for different q's measures the degree of inhomogeneity of the fractal in the sense of the probabilities associated with its subsets (mesh squares) [AtSV88], or alternatively, the degree of the *multifractality* of the fractal F.

In Eq. 2.93 and Eq. 2.94, D_q only has a physical meaning when q is an integer, however q can be any real number [HePr83]. Varying q scans through all degrees of point density existing in the fractal F since at different q, different subsets associated with different scaling indices become dominant in the determination of D_q [HJKPS86, AtSV88].

Another property of D_q is that it is a monotonically decreasing function of q, so that

$$D_a \geq D_b, \quad \text{for} \quad a \leq b \tag{2.105}$$

This is obvious if we consider that for any j, $p_j < 1.0$, and then for $q \geq 0$,

$$p_j^q \geq p_j^{q+1} = p_j p_j^q \tag{2.106}$$

$$\sum_{j=1}^{N_k} p_j^q \geq \sum_{j=1}^{N_k} p_j^{q+1} \tag{2.107}$$

$$\log\left(\sum_{j=1}^{N_k} p_j^q\right) \geq \log\left(\sum_{j=1}^{N_k} p_j^{q+1}\right) \tag{2.108}$$

From the ratios of D_{q+1} and D_q for any q,

$$\frac{D_{q+1}}{D_q} = \left(1 - \frac{1}{q}\right) \frac{\log\left(\sum_{j=1}^{N_k} p_j^{q+1}\right)}{\log\left(\sum_{j=1}^{N_k} p_j^q\right)} = (sum < 1)(ratio < 1) < 1, \quad q \geq 0 \tag{2.109}$$

There are ample proofs in the literature for $D_{HM} \leq D_I \leq D_C$ [GrPr83a, GrPr83c, HePr83, Kins94f]. For $D_q \leq D_{q+1}$ for $q \geq 0$, two sources provided a proof [Grass83a, HePr83], at least one stated the property without any proof [PeJS92], and several [Kins94f, FeKi95] provided a generalization based on $D_{-\infty} \geq D_{+\infty}$. The author is not aware of any proof extending the monotonicity property for $q < 0$. In Eq. 2.109, $(1 - 1/q) > 1$ for negative q, which means that the ratio of the logs must satisfy $ratio < 1/sum$ instead of $ratio < 1$, and it is not obvious from Eq. 2.109 whether this is true. The following informal proof is provided by the author to extend monotonicity to D_q for $q < 0$.

For $q \leq 0$, then

$$1-\frac{1}{q}=1+\frac{1}{|q|}=\frac{|q|+1}{|q|} \tag{2.110}$$

If we multiply inside the brackets of the numerator *log()* by $\left(p_{\min}^{q+1} / p_{\min}^{q+1}\right)$, then

$$\lim_{q\to-\infty}\sum_{j=1}^{N_k}\left(\frac{p_{\min}^{q+1}}{p_{\min}^{q+1}}\right)p_j^{q+1}=\lim_{q\to-\infty}p_{\min}^{q+1}\sum_{j=1}^{N_k}\left(\frac{p_j}{p_{\min}}\right)^{q+1}=p_{\min}^{q+1} \tag{2.111}$$

Substituting Eq. 2.111 into Eq. 2.110 and performing a similar operation inside the denominator *log()*, we conclude our proof with

$$\lim_{q\to-\infty}\frac{D_{q+1}}{D_q}=\left(\frac{|q|+1}{|q|}\right)\frac{\log\left(p_{\min}^{q+1}\right)}{\log\left(p_{\min}^{q}\right)}=\frac{|q|+1}{|q|}\left(\frac{q+1}{q}\right)=\frac{|q|+1}{|q|}\left(\frac{|q|-1}{|q|}\right)=1-\frac{1}{q^2} \tag{2.112}$$

This monotonicity property can be and should be employed to verify programs which compute D_q, or only D_{HM} (D_0), D_I (D_1), or D_C (D_2).

2.5.7 Generalized Correlation Dimensions

2.5.7.1 Derivation

The 2nd-order generalized *entropy* function D_2 defined by

$$D_2=\lim_{r\to0}\frac{\log\sum_{j=1}^{N_r}p_j^2}{\log(r)}=\lim_{k\to\infty}\frac{\log\sum_{j=1}^{N_k}p_j^2}{\log(r_k)} \tag{2.113}$$

can be interpreted as the *correlation* between neighbouring points on the fractal F, and can be approximated by the use of a pair correlation function C_k [GrPr83a] such that

$$D_2=\lim_{k\to\infty}\frac{\log(C_k)}{\log(r_k)} \tag{2.114}$$

where C_k is the total number of pairs of points P_n and P_m such that $d_{mn} < r_k$.

C_k can be implemented as:

1. the sum of the number of point pairs (n_j) in each *non-overlapping vel*, N_{Tk} being the total number of points covered by N_k vels [Kins94c] so that

$$C_k = \lim_{N_{Tk}\to\infty} \frac{1}{N_{Tk}^2} \sum_{j=1}^{N_k} \left(n_j^2 - n_j\right) \tag{2.115}$$

2. the sum of the number of point pairs P_i and P_j in *overlapping vels* of radius r_k centred at P_i, for each point P_i of the fractal F so that

$$C_k = \lim_{N_{Tk}\to\infty} \frac{1}{N_{Tk}} \left\{ \sum_{i=1}^{N_{Tk}} \left[\frac{1}{N_{Tk}} \sum_{j=1}^{N_{Tk}} \theta\left(r_k - \left\| P_i - P_j \right\|_2\right) \right] \right\} \tag{2.116}$$

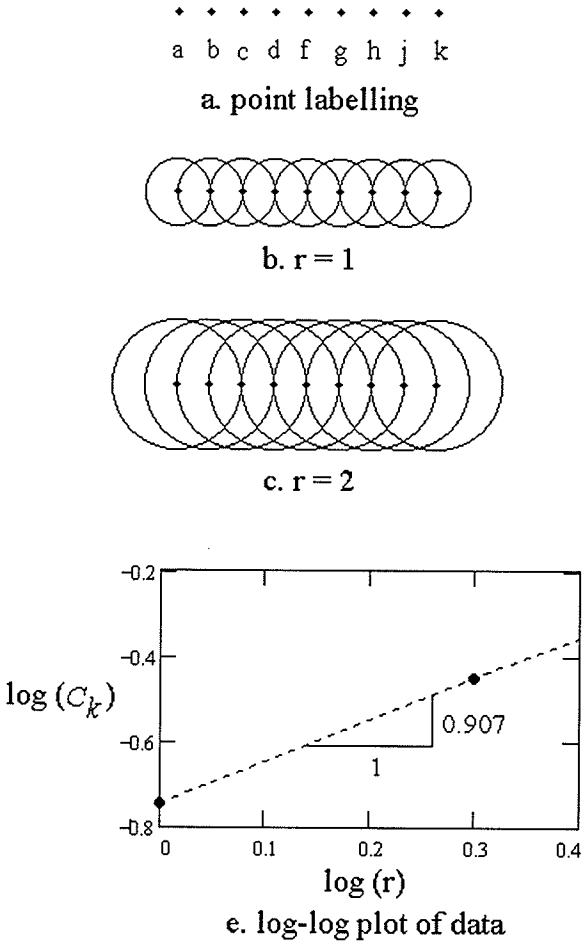
where θ() is the Heaviside step function defined by

$$\theta(z) := \begin{cases} 1 & \text{if} \quad z \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.117}$$

Equation 2.16 is the expression commonly used in literature [GrPr83b, MiPG84, AtSV88]. Kinsner has shown that both Eqs. 2.115 and 2.116 approach D_C in the limit [Kins94c] and has developed fast computational techniques for their calculation.

Figure 2.12 illustrates the procedure for computing the correlation dimension. In the example, we have a line of 9 points, labelled as shown in Fig. 2.12a. Then using vels radius of r=1, one vel centred at each of the 9 points (Fig. 2.12b), we count the number of points in each vel and record them in a table (Fig. 2.12d). This is then repeated for vels of radius r=4 (Fig. 2.12c). After all vels and all the vel sizes (here we use only 2 sizes for illustration) have been processed, the correlation function C_k is computed (Fig. 2.12d) for each scale (vel size). Finally, the correlation dimension is the slope of line which best fits

the $\log(C_k)$ versus $\log(r)$ data (Fig. 2.12e). Due to the small sample size, end effects cause the dimension of the line segment to be underestimated.



d. Numerical Results

Point i	# Points j where $d_{ij} \le r$	
	r = 1	r = 2
a	1	2
b	2	3
c	2	4
d	2	4
e	2	4
f	2	4
g	2	4
h	2	4
j	2	3
k	1	2
C_k	16/81	30/81

Fig. 2.12: Illustration of the correlation dimension.

The generalized correlation function (Eq. 2.116) has been extended by Pawelzik and Schuster [PaSc87] to the definition of generalized correlation dimensions, similar to the generalized entropy dimensions. The form for the q-order, generalized correlation function C_{q,r_k} is given by

$$C_{q,r_k} = \lim_{r_k \to 0} \left\{ \frac{1}{N_{Tk}} \sum_{i=1}^{N_{Tk}} \left[\frac{1}{N_{Tk}} \sum_{j=1}^{N_{Tk}} \theta\left(r_k - \left\| P_i - P_j \right\|_2\right) \right]^{q-1} \right\}^{1/(q-1)} \tag{2.118}$$

which can be used to calculate the Rényi dimensions as

$$D_q = \lim_{r \to 0} \frac{\log\left(C_{q,r_k}\right)}{\log(r)} \tag{2.119}$$

2.5.7.2 Properties of D_q

The generalized correlation dimensions measure the scaling properties of the density of points on the fractal F. Since the generalized correlation dimensions use overlapping vels, their computed dimensions D_q will be slightly greater than those calculated using the generalized entropy dimensions which use non-overlapping vels [Kins94c, p. 8]. Correlation-based dimensions require significantly less memory and are easier to calculate than entropy-based dimensions, especially for objects having $D > 2$ [GrPr83a], and have been shown to *execute* much faster [AtSV88]. Additionally, generalized correlation methods seem to *converge* to D_q much faster while using fewer points than box counting methods [GrPr83a, p. 348]. Correlation methods are also well suited for the analysis of temporal signals [GrPr83b, MiPG84].

2.5.7.3 Accuracy

Stacey measured the accuracy of the correlation dimension on Koch curves for which the dimension is calculable from the self-similarity of the curves, and found the error to be at least 1.6% [Stac95]. Since this has been done previously, it was not done as part of this thesis.

2.5.8 Mandelbrot Dimension

The basic concept of fractal dimensions is that some measure such as length, area, or density of the fractal exhibits scaling properties. The Mandelbrot dimension, also known as the multifractal spectrum or the spectrum of singularities, can be used to characterize the global scaling properties of the fractal F [AtVS88] for which the *distribution of the measure* itself satisfies a multi-valued power law such that

$$p_j\left(r_j\right) \propto r_j^{\alpha_j} \tag{2.120}$$

where p_j is the probability for the j^{th} vel of size r, and α_j is the strength of the local singularity of the measure [Kins94f, p. II-95]. In other words, the Mandelbrot dimension probes the probability distribution *right inside the vels* we use to obtain D_q.

2.5.8.1 Derivation of D_{MAN}

The Mandelbrot dimension D_{MAN} [Kins94f], , is expressed in terms of

- α_q which describes the strengths of the singularities of the scaling properties, also referred as [PeJS92] the local Hölder exponent, singularity index, and singularity strength.
- $f(\alpha)$ which describes the density distribution of the singularities [HJKPS86].

α_q and $f(\alpha)$ are obtained from D_q by a straightforward Legendre transformation [PaSc87]. The exact derivations of α_q and $f(\alpha)$ are presented elsewhere [HJKPS86, Chen95, ChKi95], the results of which yield

$$\alpha_q = \frac{d}{dq}\left[(q-1)D_q\right] \tag{2.121}$$

$$f(\alpha) = q\alpha_q - (q-1)D_q \tag{2.122}$$

If we substitute Eq. 2.94 into Eq. 2.121 and solve, then

$$\alpha_q = \frac{\sum_{j=1}^{N_k} n_j^q \log\left(\frac{n_j}{N_k}\right)}{\log(r_k)\sum_{j=1}^{N_k} n_j^q} \tag{2.123}$$

Substituting Eqs. 2.94 and 2.123 into Eq. 2.122 yields an expression more complex than Eq. 2.122, so we let Eq. 2.122 describe the simplest calculation of $f(\alpha)$. D_q can be expressed in terms of α_q and $f(\alpha)$ [HJKPS86] as

$$D_q = \frac{1}{q-1}\left[q\alpha_q - f(\alpha_q)\right] \tag{2.124}$$

2.5.8.2 Special Values of D_{MAN}

It can be shown from Eqs. 2.94, and 2.121 through 2.123, that [PeJS92, Kins94f]

$$D_{-\infty} = \alpha_{-\infty} = f_{-\infty}, \quad q = -\infty \tag{2.125}$$

$$D_{+\infty} = \alpha_{+\infty} = f_{+\infty}, \quad q = +\infty \tag{2.126}$$

$$D_0 = f_0 = \max\left(f(\alpha_0)\right), \quad q = 0 \quad \text{(see Fig. 2.13)} \tag{2.127}$$

$$\alpha_1 = f_1, \quad q = 1 \tag{2.128}$$

2.5.8.3 Properties of D_{MAN}

Each singularity strength, or Hölder exponent, α has its own dimension $f(\alpha)$ [HJKPS86]. α, also called the crowding index [Kins94f], provides a measure of the density for different subsets of the fractal F: small values of α characterize the dense subsets of F, and large values of α characterize the rarefied subsets, and α_{max}-α_{min} can be used as a measure of the inhomogeneity of the fractal [AtSV88]. For pure fractals, (ie.

homogeneous, or non-multifractals), $\alpha_{max} = \alpha_{min}$, and the Mandelbrot dimension curve reduces to a single point (see. Fig. 2.13).

The function $f(\alpha)$ has the property [PeJS92]

$$\frac{df(\alpha)}{d\alpha} = q \tag{2.129}$$

which combined with Eq. 2.128 yields the property that at α_1, $f(\alpha)$ contacts the bisector defined as $f(\alpha) = \alpha$ (see Fig. 2.13).

Figure 2.13 contrasts the Mandelbrot dimensions of a homogeneous (fractal) object and a non-homogeneous (multifractal) object.

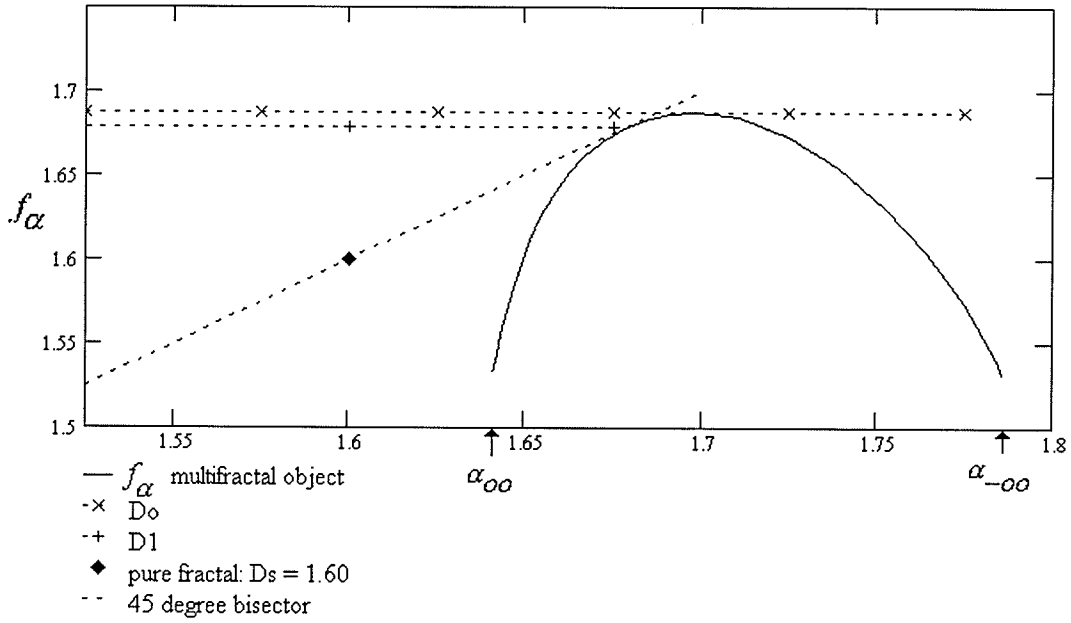


Fig. 2.13: Mandelbrot dimensions of 2 objects.

2.5.9 Variance Dimension

Kinsner [Kins94b, GrKi95] has developed a variance-based dimension measure D_σ which is ideally suited to analyse temporal signals in real-time, and from which the spectral dimension D_β can be derived in a trivial manner. The variance dimension

measures the temporal scaling properties of the variability or variance of the signal amplitude of some signal *x(t)*. The values D_σ assumes with sampling time δt is known as the trajectory of D_σ.

2.5.9.1 Derivation

The variance, σ^2, of the amplitude increments of signal *x(t)* over a discrete sampling time increment Δt conforms to the power law

$$\sigma^2\left[x(t_{1+\Delta t})-x(t_1)\right]=\sigma^2(\Delta x(\Delta t))\sim|\Delta t|^{2H} \tag{2.130}$$

where H is the Hurst exponent associated with fractional Brownian motion, and can be determined from

$$H=\lim_{t\to\infty}\frac{1}{2}\frac{\log\left(\sigma^2(\Delta x(\Delta t))\right)}{\log(\Delta t)} \tag{2.131}$$

The variance dimension D_σ [Kins94b] can be obtained from

$$D_\sigma=2-H \tag{2.132}$$

and the spectral dimension can be determined [Kins94b] in real time from

$$\beta=2H+1 \tag{2.133}$$

2.5.9.2 Properties

The variance dimension is acutely sensitive to low-level noise contaminating the signal of interest, *x(t)*. It has been used in speech processing to establish phonemic boundaries based on the trajectory of D_σ and to characterize the bounded phonemes. The behaviour of the variance dimension is controlled by several factors [Kins94b]

1. the *sampling window size* which controls the sensitivity of D_σ to $\Delta x(t)$,

2. the *overlap* between successive sampling windows which has the effect of low-pass filtering the trajectory of D_σ, and
3. the *probing interval* [GrKi95]:
    - unit increments $\{1\delta t, 2\delta t, 3\delta t,...\}$ which provide high separation of signal and noise, and
    - dyadic increments $\{2\delta t, 4\delta t, 8\delta t,...\}$ which provide high amplification of the dimension trajectory at the expense of high signal and noise separation.

It should be stated here that the variance dimension fails for signals of constant value such as $x(t) = k$, where k is some constant, since the variance is 0.0, and

$$\log(0.0) = -\infty \qquad (2\text{-}134)$$

2.6. Summary

This chapter has covered an enormous swath of material: from the modelling of Laplacian fractals, to the numerical techniques available to efficiently and quickly solve Laplace's equation for the potential field paying attention to the moving boundaries in the mesh's interior, and finally concluding with multifractal measures at our disposal to qualitatively measure, analyse, and compare the Laplacian fractals. The following chapter describes the software which was designed and implemented, based on the theory presented in this chapter, to model, grow, and measure Laplacian fractals while allowing control over as many parameters as possible.

CHAPTER 3

SYSTEM IMPLEMENTATION

3.1. Introduction

To test, validate, refine and implement concepts and algorithms discovered and developed during the course of the research for this thesis, several major programs were written, debugged, and rigorously tested.

In order to:

- capitalize on the advantages of an object-orientated programming language,
- capitalize on the encapsulation and abstraction provided by Borland's Object Windows Library (OWL®),
- capitalize on acquired software tools and previous Windows programming experience,
- allow transparent manipulation of data structures exceeding 64 Kb without any modifiers such as *huge*, and the use of 32-bit array indexes,
- allow pasting of contents of informational windows into Windows-based word processors, and
- allow seamless software development at different locations: at the university, place of employment, and at home,

the main programs were written in C++ and targeted for the Microsoft Windows environment for personal computers, specifically the Win32 implementation of Windows. Simple auxiliary programs to test some hypotheses were coded for execution in a DOS

environment for cases where data was saved to disk files or a graphical display of data was not needed. These validated concepts were then integrated into the main programs.

The three main programs are *Random.exe*, *Methods.exe* and *Corona32.exe*. *Random.exe* was written to evaluate the pseudo-random and chaotic generators. It allows the user to individually analyze any one of 16 different generators. *Methods.exe* was written to test, debug, validate, and evaluate the various point, line, and block iterative algorithms. *Corona32* was written to:

- recreate some of the fractals generated by G. Stacey [Stac95],
- test the effects of random generators on Laplacian fractals,
- test the effects of the iterative algorithms on Laplacian fractals,
- allow a user to create and save simulation cases, having full control over
    - ◊ the electrode topology and parameters,
    - ◊ some of the parameters of the dielectric breakdown model,
    - ◊ the random generator used,
    - ◊ the iterative solution methods for grid initialization, and local and global updates,
    - ◊ the convergence criteria, and
    - ◊ the format of the informational windows.
- allow the user to setup batches of cases to run unattended, saving the results after each simulation run is completed,
- present as much useful information about the simulation dynamics as possible during the run,

- measure the fractality and multifractality of generated Laplacian fractals, and
- validate the routines used to measure the multifractality.

The software was constructed in an object *class*-orientated style wherever possible to "group" related operations and data structures into portable and re-useable functional ***class***es. This separation of the code into classes aided immensely in the testing and debugging of code, the implementation of new features, maintainability and understandability of existing code, and the incorporation of code validated in other programs. *Corona32.exe* incorporates the validated random and generator classes from *Random.exe* and the validated iterative solution classes from *Methods.exe*.

Appendix F [Aren96] contains the source code unique to *Random.exe*, Appendix G [Aren96] contains the source code unique to *Methods.exe*, and Appendix H [Aren96] contains the source code for *Corona32.exe*, some of which is common to *Random.exe* and *Methods.exe*.

3.2. Taxonomy of Classes

3.2.1 Introduction

It may be helpful to consider the taxonomy of object classes coded and utilized by the programs developed during the course of this research. Not all programs utilize all these types of classes. The basic taxonomy shown in Fig. 3.1 and described below is related to the *relationships* between classes, such as parent, child, and sibling classes and the *utilization* of classes, not the *inheritance* associated with classes.

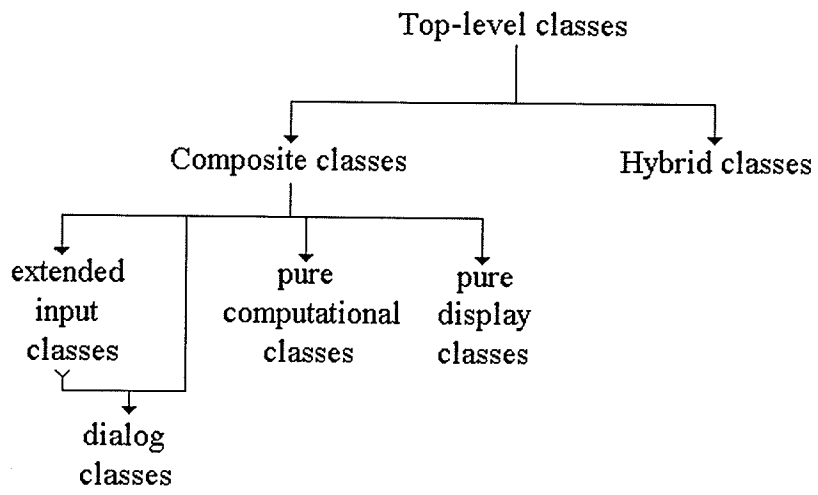


Fig. 3.1: Taxonomy of classes.

3.2.2 Dialog Classes

Dialog classes are the simplest of all classes. They essentially consist of the dialog constructor function which instantiates the dialog controls in the dialog box. The Borland C++ Object Windows Library (OWL) classes handle the transfer of initialization data to, and the user entered/selected data from, the dialog controls.

3.2.3 Extended Input Classes

Extended input classes utilize one or more input dialog classes, but additionally provide error checking on the extracted, returned data and may possibly force the user to enter correct data or choose default data before proceeding.

3.2.4 Pure Computational Classes

Pure computational classes perform specific computational functions on *supplied* data objects or on *global* data objects. These classes do not call any display classes or display functions. Some classes and functions just modify global objects. Others return

computed results to the caller upon completion, or store the results for later recall. Pure computational classes can utilize other computational classes without falling into the composite class category.

3.2.5 Pure Display Classes

Pure display classes just format and display global or received data objects or structures, most commonly in graph or scrolling chart styles. No mathematical computations other than scaling and offsetting are performed.

3.2.6 Hybrid Classes

Hybrid classes process *and* display received or global data objects. Computed results may be returned to the caller or stored for later recall.

3.2.7 Composite Classes

Composite classes route incoming, received data to one or more computational classes, extract the computed results, and route the results to one or more display classes. These classes use computational *and* display classes as their basic building blocks.

3.3. Study of Random and Chaotic Generators

3.3.1 Purpose

Random.exe was written to evaluate the pseudo-random and chaotic generators. It allows the user to individually analyze any one of 16 different generators. Each generator is implemented as a distinct object class with constructor and destructor functions, *Randomize()*, *Rand ()*, and *Rand (MaxValue, MinValue)* functions. The last function

allows the generators output to be scaled to any interval. The implementation of random generators as classes allows multiple, non-interfering instances of the same generator, and allows the data members for each generator to be kept separate from those of other generator classes. When a generator is selected, its class object is instantiated, and the first 100,000 numbers in its sequence are obtained and stored in an array as $x(t)$. The chaotic generators use at least 2 independent variables, so their $y(t)$ sequence is also obtained and saved. The range of these numbers is divided into 1000 equal-sized intervals, and the distributions of $x(t)$ and $x(t)$-$x(t-1)$ are computed and saved to a file, and where applicable, the same is done for $y(t)$ and $y(t)$-$y(t-1)$. Some statistics of $x(t)$, $x(t)$-$x(t-1)$, $y(t)$, and $y(t-1)$ such as the minimum, maximum, mean, standard deviation and the variance of the original sequences are computed and saved in a file. This is then repeated for $x(t)$ and $y(t)$ normalized to reside in the interval of

$$0.0 \leq x(t), y(t) \leq 1.00 \tag{3.1}$$

The user is then prompted

- if the sequence of $x(t)$ and $y(t)$ should be saved to a file, and
- if the variance dimension should be performed on $x(t)$, and
- if $x(t)$ and $y(t)$ should be analyzed for duplicate values and sequences.

3.3.2 Class Structure

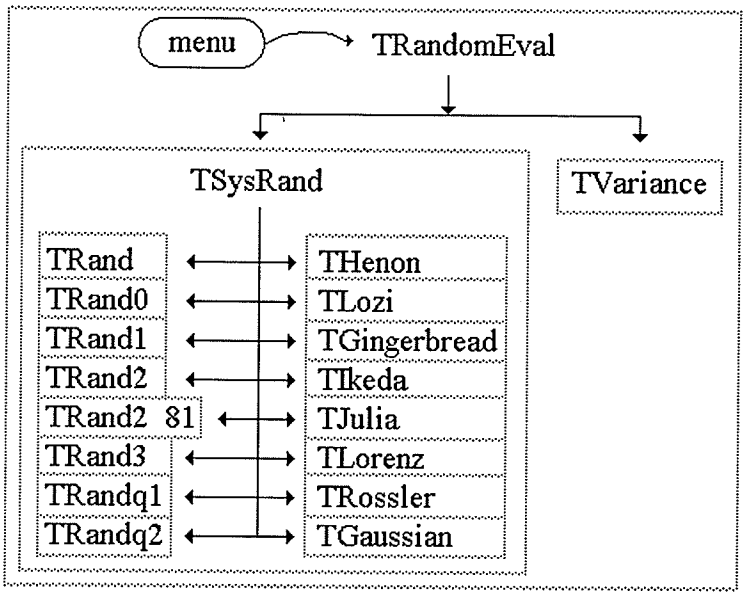


Fig. 3.2: Relationships of classes in *Random.exe*.

3.3.3 Class Descriptions

All the classes employed by *Random.exe* are pure computational classes.

TRandomEval

TRandomEval instantiates the TSysRand and TVariance computation classes and manages them while executing the selected menu commands. All menu commands are processed by this class. The obtained results are saved to files or presented in message boxes.

TVariance

TVariance is used by TRandomEval to compute the variance dimension of the $x(t)$ sequence.

TSysRand

TSysRand is a container, encompassing, or wrapping class. It manages 16 random and chaotic generators, hiding their implementation details from the caller. TSysRand manages the selection, creation and deletion of instances of generators, and presents a single unified interface to the caller via the standard *Randomize()*, unscaled *Rand()*, and scaled *Rand(MaxValue, MinValue)* functions through which the parent of TSysRand can indirectly access the selected generator.

3.3.4 Validation

The output files produced by analyzing a generator called *RANDX* are:

- *RANDX*.dat - statistical measures of *x(t)*, *x(t)-x(t-1)*, *y(t)*, and *y(t)-y(t-1)*,
- *RANDX*.prn - distributions *x(t)*, *x(t)-x(t-1)*, *y(t)*, and *y(t)-y(t-1)*,
- *RANDX*d.prn - sequence of *x(t)* and *y(t)*,
- *RANDX*ds.prn - variance dimension of *x(t)*.

The *RANDX*d.prn files were loaded into Mathcad®, and the proper generation of the chaotic attractor coordinates were verified by plotting *x(t)* versus *y(t)*. To validate the statistical measures, the built-in mean, maximum, minimum, standard deviation, and variance functions were applied to the *x(t)* and *y(t)* sequences and the results compared with those in the corresponding *RANDX*.dat file.

3.4. Study of Iterative Methods

3.4.1 Purpose

Methods.exe was written to test, debug, validate, and evaluate the point, line, and block iterative algorithms. It allows the user to:

1. measure the time required for each method to complete 100 global sweeps on meshes of size
    - 125×125,
    - 250×250, and
    - 500×500

    for a parallel plate capacitor, and having
    - no singularities, and
    - singularities at approximately 10% of all mesh points [to emulate corona], and,

    thus determining the average solution time for each implemented method
2. to abort the timing measurements as they can take several hours to complete,
3. to test the "injection" or introduction of singularities at approximately 10% of all mesh points for the 3 mesh sizes, and
4. to verify the method of lines for short and long segments with various boundary conditions.
5. to verify the stencils for accelerated and non-accelerated block methods with various boundary conditions along the mesh edges and all possible combinations of enclosed singularities within the blocks.

6. to measure the optimal acceleration factor ω_{opt} for various square and rectangular shaped meshes.

3.4.2 Class Structure

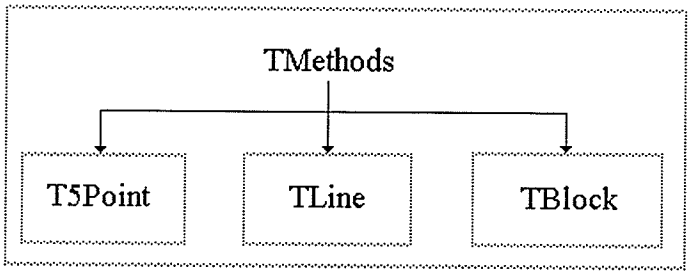


Fig. 3.3: Relationship of classes in *Methods.exe*.

3.4.3 Class Descriptions

All the classes employed by *Methods.exe* are pure computational classes.

TMethods

TMethods instantiates the iterative computation classes and manages them while executing the selected menu commands. The obtained results are saved to files or presented in message boxes.

T5Point

T5Point implements the 5-point operator-based point iterative methods. Non-accelerated Gauss-Seidel type and accelerated SOR, normal row-wise scanning and alternating direction (row-wise followed by column-wise) scanning techniques are implemented. Any combination of enclosed singularities and Neumann boundaries at the left and right side endpoints (row segments) or along the left or right columns (column segments) are supported.

TLine

TLine implements the line iterative methods. Non-accelerated (Gauss-Seidel type) and accelerated, normal row-wise scanning and alternating direction (row-wise followed by column-wise) scanning techniques are implemented. Any combination of enclosed singularities and Neumann boundaries at the left and right side endpoints (row segments) or along the left or right columns (column segments) are supported.

TBlock

TBlock implements the block iterative methods using 2×2 blocks of points. Non-accelerated (Gauss-Seidel) and accelerated, normal row-wise scanning and alternating direction (row-wise followed by column-wise) scanning techniques are implemented. Any combination of enclosed singularities and Neumann boundaries along the left and right side are supported.

3.4.4 Validations

The proper segmentation of lines to exclude singularities was manually verified for various locations of singularities along a test line of 10 points. Particular attention was paid to the segmentation for singularities located 1 and 2 points away from Neumann and Dirichlet boundaries at the endpoints. A routine to test the segmentation and solution of short row and column segments of 1, 2, 3, and 4 points existing within 10-point rows and columns for all possible boundary conditions was written and the results were saved to a file. These results were verified by using Mathcad to generate the proper matrix inverses for short line segments as well as for unsegmented 10-point line segments, for the identical boundary conditions. Assurance of correct line and block computational

methods was considered essential and crucial to the acceptance of the results of the multifractal analysis of the fractals generated by these methods.

3.5. Laplacian Fractal Simulation Program

3.5.1 Classes

3.5.1.1 Display classes

TChart

The TChart class permits the graphing of data in a left-scrolling, chart recorder-like window utilizing the following styles:

- vertical bars (as in bar charts)
- connected lines (effective linear interpolation between data points), or
- points indicated by "×"'s,

using selectable red, blue, or magenta colours. The chart scrolls to the left by about 25% of the chart window width whenever plotted data touches the right side of the chart. Scrolling occurs at discrete steps, and not continuously, so as to reduce the time spent redrawing the objects displayed in the window. A TChart instance only supports one style, so effectively, it can only be used to display a single function. Data to be plotted is passed to a TChart instance, one piece at a time, additionally specifying its desired colour.

TLGraph

TLGraph permits the graphing of data utilizing the following styles:

- connected lines (effective linear interpolation between data points), or

- points indicated by "×"'s,
- points indicated by "×"'s *and* connected by lines.

Data to be plotted is stored in a **new**'ed (dynamically allocated) block of memory by the calling function, with one curve per memory block. The address of this block, the number of data points it contains, and the desired curve style are passed to TLGraph. The calling function relinquishes all rights to the block. The TLGraph destructor function is responsible for **delete**'ing (**free**'ing) all collected blocks. If the block of data actually specifies just a single point, then a single "×" is plotted. TLGraph can be used to plot multiple functions on one graph.

TField

TField provides a window which contains a gray-shaded picture of the potential field. The maximum potential is divided into 10 intervals, each represented by a black-to-white band.

If the original mesh is less than 512×512, then for display purposes, TEnlarge is used

- to increase the mesh resolution by some integer magnification factor, and then
- to use Laplacian interpolation followed by point Gauss-Seidel iteration for all the new mesh points.

The class organization within TField is shown within Fig. 3.4.

TCoronaView

TCoronaView displays a black-on-white-background image of the Laplacian discharge fractal (i.e. corona discharge).

TFieldView

TFieldView is a composite, pure display class. It creates a window which contains a black-on-white image of the Laplacian fractal using TCoronaView, and a gray-shaded picture of the potential field using TField. The pictures appearing in Appendix E are saved screen captures of TFieldView windows. The class structure of TFieldView is shown in Fig. 3.4. TCorona can be selectively updated, and is usually done so after every 10^{th} new corona cell. TField is generally updated only after the Laplacian fractal is complete, yielding the final field. Intermediate updates of the field image are too costly time-wise, and are generally avoided. The TFieldView window contents can be saved to a *.bmp file.

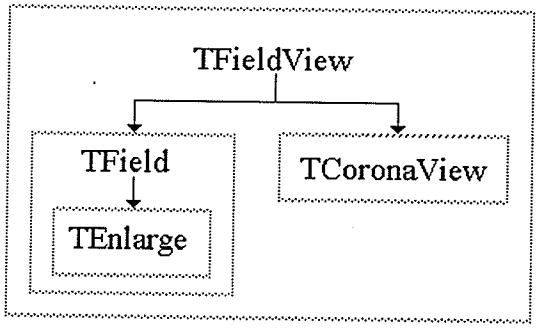


Fig. 3.4: Relationships of classes in TFieldView.

TGrowth

TGrowth provides more information than TCoronaView. The corona cells, neighbouring candidate cells, and zero-potential cells are all colour-coded. This class was developed to test the extraction of the candidate cells and their maintenance in a candidate list.

TErrorContour

TErrorContour displays the mesh residuals in a colour-coded scheme. It is useful for monitoring the wavefront-like propagation of corrections for the various iterative methods, and for observing the location and trajectory of the maximum residual.

3.5.1.2 Computational classes

TKoch

This class is used to generate a test fractal having a known self-similarity dimension of $D_S = 1.5$. This fractal was chosen for 2 reasons:

- it has a "nice" value for its dimension (4/3 = 1.333... is *not* nice), and
- its points can be described by integer coordinates, allowing the dimension algorithms designed for mesh-based Laplacian fractals to be employed for this fractal as well.

The fractal is produced by recursively applying a generating algorithm to an initiator. In this case, the initiator is a square, and at each stage, the generator replaces each original segment of length r by an 8-segment curve (Fig. 3.5) whose segments are each of length $r/4$. The first and second iterations of this Koch curve are shown in Figs. 3.6 and 3.7.

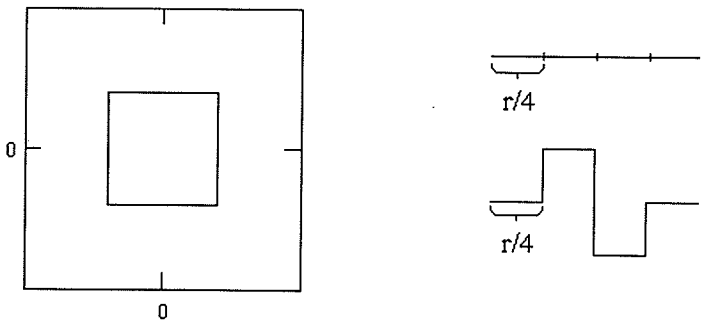


Fig. 3.5: Koch initiator (left) and generator (right).

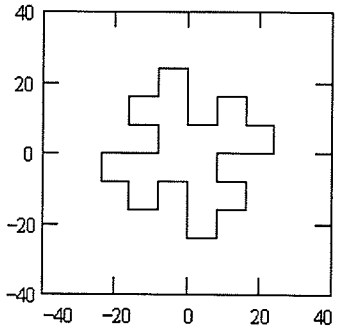


Fig. 3.6: 1st stage of Koch curve

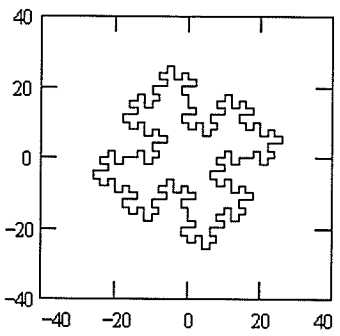


Fig. 3.7: 2nd stage of Koch curve.

The relation between the reduction factor s and the number of pieces N into which a self-similar structure can be subdivided is described as [PeJS92, p. 205]

$$a = \frac{1}{s^D} \tag{3.2}$$

and the self-similarity dimension D_S is defined as

$$D_S = \frac{\log a}{\log 1/s} \tag{3.3}$$

Since each scale reduction of $s = 1/4$ yields 8 new segments, the self-similarity dimension of this Koch curve is

$$D_S = \frac{\log 8}{\log 4} = 1.50 \tag{3.4}$$

TEnlarge

Allows the enlargement of small meshes using dyadic magnification scales so that the fine field details can be magnified without suffering from pixelation. TEnlarge magnifies the mesh resolution as follows:

1. a temporary mesh of the desired size is created,
2. the potential values of the existing mesh are transferred to the new, large mesh, being

properly spread out (decimated).

3. *Original* Dirichlet boundaries are properly propagated on all new mesh cells inserted between existing cells with Dirichlet boundary conditions. All the transferred values are known, so from this point on considered they are treated as Dirichlet boundaries and are not updated. The object from this point on is to compute the potentials of all the new, added mesh points.
4. For each effective, successive doubling of the mesh resolution, Laplacian interpolation is performed at all new mesh points at that scale, followed by 10 global passes of point Gauss-Seidel (for new points only). Since the actual size of the new mesh is fixed, this is accomplished by varying the arms and the rotation of the 5-point operator stencil. The process of mesh resolution doubling, interpolation, iteration is continued until the desired magnification has been attained.

TStability

TStability measures the divergence and stability of some variable *x(t)*, usually the infinite norm. In iterative processes of the type used in this research, the trajectory of the maximum norm may actually stall. In other words, it may stop decreasing and remain constant, preventing convergence due to that criterion, and experiments show that any further iterations will not decrease the value of *any* norm. TStability can be utilized to detect this situation prior to performing the maximum number of iterations. *x(t)* is considered to be stable if the difference of the n^{th} value of *x(t)* from the running average is within *k%* of the average for *m* consecutive iterations, such that

$$\left|x(n) - ave(x(t))\right| < k \times ave(x(t)) \tag{3.5}$$

x(t) is considered to be diverging if at any time the difference of the n^{th} value of *x(t)* from

the running average exceeds $d\%$ of the average, such that

$$\left|x(n) - ave(x(t))\right| \geq d \times ave(x(t)) \tag{3.6}$$

In practice d must be much greater than k. If we assume that the maximum eigenvalue λ_{max} is dominant and that the effects of other eigenvalues have decayed sufficiently, then at any time n, the previous n values of the infinite norm can be represented by the power series

$$a + a\lambda_{\max} + a\lambda_{\max}^2 + \ldots + a\lambda_{\max}^{n-1}, \quad \text{whose average is} \quad ave = \frac{a\left(1-\lambda_{\max}^n\right)}{n\left(1-\lambda_{\max}\right)},$$

and the most recent value by λ^n_{max}. Then it can be shown that the maximum spread between $x(n)$ and the average is

$$\left(\frac{ave(x(t)) - x_n}{ave(x(t))}\right) \times 100\% = \left(1 - \frac{n\lambda_{\max}^n\left(1-\lambda_{\max}\right)}{\left(1-\lambda_{\max}^n\right)}\right) \times 100\% \tag{3.7}$$

If $\lambda_{max} = 0.99$ and $n = 10$ the spread is 5.4%, and for $n = 100$ the spread is 42.3%. Thus for this example, a suitable value of k to detect a stalled infinite norm would be $k \leq 1.0\%$.

TLinear

TLinear calculates the slope of the best fitting line of a list of n data points or 2 vectors containing the x and y coordinates of n data points using linear regression.

TQuadratic

TQuadratic calculates the coefficients a, b, and c for the equations

$$y(x) = ax^2 + bx + c, \quad \text{or} \quad x(y) = ay^2 + by + c \tag{3.8}$$

based on 3 supplied points. The slope at any point (x,y) can then be calculated using

$$\frac{d}{dx}y(x) = 2ax + b, \quad \text{or} \quad \frac{d}{dx}x(y) = \frac{1}{\frac{d}{dy}x(y)} = \frac{1}{2ay + b} \tag{3.9}$$

TEqns

TEqns solves

$$Ax = b \tag{3.10}$$

for case of A being a 4×4 or an N×4 matrix. If A is square, the solution for x is given by

$$x = A^{-1}b \tag{3.11}$$

If $N > 4$, then A is not square and is considered to be over-determined, and the least squares approximation solution for x is given by

$$x = \left(A^T A\right)^{-1}\left(A^T b\right) \tag{3.12}$$

TCubic

TCubic calculates the *maximum* slope of the best fitting cubic polynomial using least squares approximations and given 2 vectors containing the x and y coordinates of n data points. TCubic attempts to determine the optimal coefficients a, b, c, and d for

$$x(y) = ay^3 + by^2 + cy + d \tag{3.13}$$

by constructing the matrices A, z, and k such that

$$A = \begin{vmatrix} y_0^3 & y_0^2 & y_0 & 1 \\ y_1^3 & y_1^2 & y_1 & 1 \\ y_2^3 & y_2^2 & y_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ y_{N-1}^3 & y_{N-1}^2 & y_{N-1} & 1 \end{vmatrix}, \quad z = \begin{vmatrix} a \\ b \\ c \\ d \end{vmatrix}, \quad k = \begin{vmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{vmatrix} \tag{3.14}$$

so that

$$Az = k \tag{3.15}$$

TEqns is utilized to compute the inverse of A required to obtain the values for the coefficients a, b, c, and d. The maximum slope occurs when the second derivative of $x(y)$ is 0 at some point $y=y_{critical}$, then

$$x''\left(y_{critical}\right) = 6ay_{critical} + 2b = 0 \quad \rightarrow \quad y_{critical} = \frac{-b}{3a}, \tag{3.16}$$

and the maximum slope is found by

$$\frac{dy}{dx} = \frac{1}{x'\left(y_{critical}\right)} = \frac{1}{3a\left(\frac{-b}{3a}\right)^2 + 2b\left(\frac{-b}{3a}\right) + c} = \frac{3a}{3ac - b^2} \tag{3.17}$$

When computing the box counting dimension D_{HM}, the resultant plot of $\log(N_k)$ versus $\log(1/r_k)$ plot is generally a monotonically increasing, "s"-shaped curve and the slope of interest is the maximum slope. TCubic was initially used to fit a single-valued cubic curve to the above plot, and then to extract the maximum slope using Eq. 3.17. Since the maximum slope usually occurs at only one point, the computed dimension is essentially determined by only one of the many scales measured. TLinear is preferred over TCubic to compute the slope since TLinear involves measurements of N_k from most, if not all scales.

T5Point, TLine, TBlock

These classes have been described previously in Sec. 3.4.3.

TSolve

TSolve (Fig. 3.8) is a computational class which wraps T5Point, TLine, and TBlock into a single iterative solution class, managing the creation and deletion of required instances. TSolve presents a unified interface for local and global iterative updates of the mesh potentials. The caller of a TSolve class is responsible to determine

and control the acceleration factor for accelerated iterative methods and to evaluate the state of convergence.

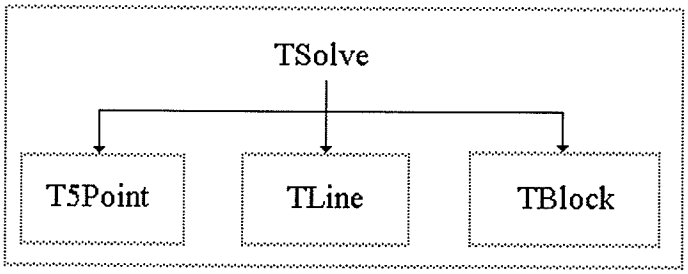


Fig. 3.8: Relationship of classes in TSolve.

TAccelSolver

This class (Fig. 3.9) supports near-optimal, dynamic acceleration of the SOR, SLOR, and SBOR accelerated iterative techniques and their derivatives. For local updates, TAccelSolver uses a local update region (LUR) of 51×51 points, centred on the most recently added corona point, and performs a fixed number of iterations without any convergence tests since convergence will not be attained within the allotted 50 local iteration sweeps anyway. Four singular-free regions, one anchored at each of the LUR corners (see Fig. 3.10), are used to estimate the spectral radius of the entire LUR (see Sec. 4.8.3) prior to any iterations. Since the position and characteristic of the LUR change with each new corona point, this simple segmentation technique is deemed sufficient.

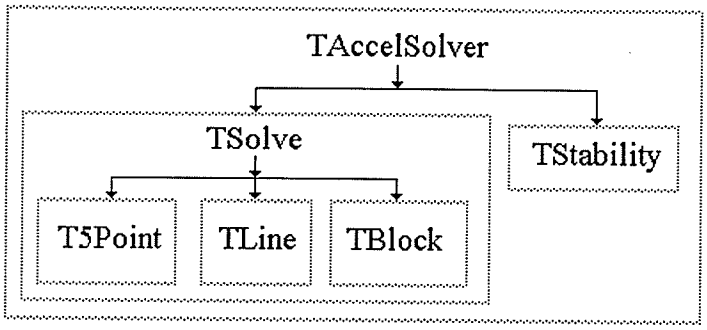


Fig. 3.9: Relationships of classes in TAccelSolver.

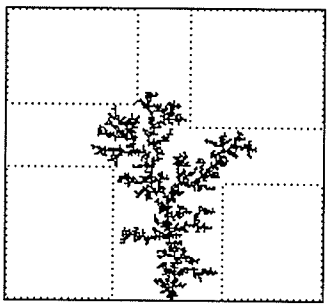


Fig. 3.10: Singularity-free regions used by local updates.

For global updates, 8 contracting singular-free regions are maintained: 4 which are anchored to the corners of the mesh (Fig. 3.11a), and 4 which are anchored to the sides of the mesh (Fig. 3.11b & c, note: region anchored to bottom side has vanished). After each new additional corona point, the extents of these regions are adjusted (effectively reduced) as needed to exclude all singular points. The directions in which the singular-free regions contract are indicated by arrows. The maximum spectral radius and the optimal acceleration factor are estimated from these regions, dependent on the selected iteration method. Optionally, the caller can specify a fixed acceleration factor.

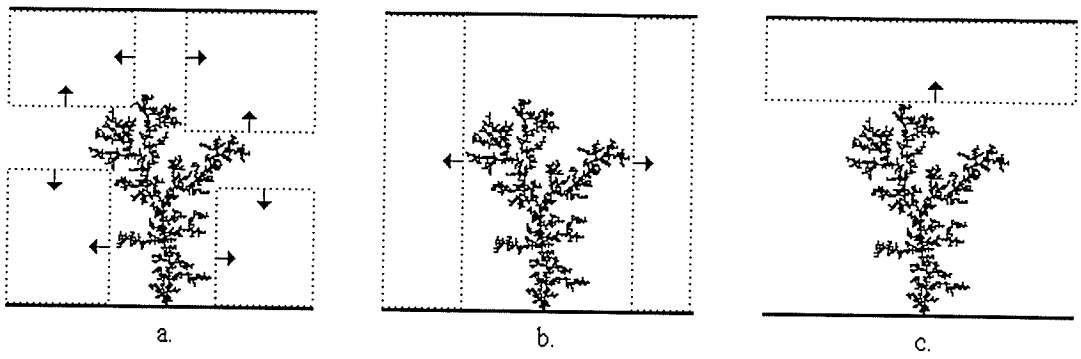


Fig. 3.11: Typical singularity-free regions used by global updates.

The specified global passes can be executed

- uninterrupted, or
- 1 pass at a time followed by convergence tests or the collection of the residual norms.

The tests for global convergence involve tests for

- complete convergence: $\| \mathbf{r}^k \|_\infty = 0$,
- near convergence: $\| \mathbf{r}^k \|_\infty \leq \textit{allowable error}$,
- instability due to over-acceleration, any $U_{i,j} \geq 1.5 \times \text{max. electrode potential}$,
- stalled convergence, utilizing TStability.

For analytical solutions of Laplace's equation, the potentials of all points are confined to the range of the boundary potentials. An artifact of using iterative methods to solve Laplace's equation is the continuous propagation of the effects of all the boundaries, leading to potentials which at some points are negative, or exceed the maximum boundary potential *MaxV*. Observations showed that the maximum mesh potential was less than 1.3×*MaxV*, hence mesh potentials exceeding 1.5×*MaxV* were considered indicative of an oscillatory or diverging trend. When instability due to over-acceleration occurs, the potentials of all non-boundary mesh points tend to increase

rapidly toward $+\infty$. To avoid the time-consuming task of checking *all* the mesh points for this condition, only the mesh points described by (16*i*, 16*j*) are checked for meshes greater than 100×100. Thus for a 512×512 mesh which has 262,144 points, only 1024 points are checked. For informational and comparative purposes, the 1st and infinite residual norms, the estimated spectral radius λ_{max}, and estimated acceleration factor ω_{est} can be returned to the calling function. TAccelSolver utilizes a TSolve class to achieve a unified interface to the large number of implemented iterative methods.

TRand, TRand0, TRand1, TRand2, TRand2_81, TRand3, TRandq1, TRandq2, THenon, TLozi, TGingerbread, TIkeda, TJulia, TLorenz, TRossler, TGaussian

Each generator is implemented as a distinct object class with constructor and destructor functions, and *Randomize()*, *Rand ()*, and *Rand (MaxValue, MinValue)* functions, similar to the standard C language *rand()* and *randomize()* functions. The last *Rand(...)* function allows the generators' output to be scaled and offset. The implementation of random generators as classes allows multiple, non-interfering instances of the same generator, and allows the data members for each generator to be kept separate from those of other generator classes.

TSysRand

TSysRand is a container, encompassing, or wrapping class. It manages the 16 random and chaotic generators, hiding their implementation details from the caller. TSysRand manages the selection, creation and deletion of instances of generators, and presents a single unified interface to the calling functions via the standard *Randomize()*, *Rand()*, *Rand(MaxValue, MinValue)* functions through which the parent of TSysRand can indirectly access the selected generator.

TVariance

TVariance computes the variance dimension D_σ given a long sequence of values. The window size, and window shift size are specified when the constructor is called. The probing increments are automatically determined. Unit increments are utilized where possible so as to attain a high separation between the signal and any contaminating noise (see Sec. 2.4.9.2).

TBoxCount

TBoxCount computes the $\log(N_k)$ and $\log(1/r_k)$ values for the Hausdorff mesh counting dimension D_{HM} (see Sec. 2.4.4). The caller of TBoxCount must supply either

- the range of scales to probe, *and*
- an array of points, or
- the name of a *.cdf or *.pnt file containing points

which lie on the fractal F being measured. TBoxCount (Fig. 3.12) computes the best fitting slope using linear regression and the maximum slope of the best fitting cubic curve.

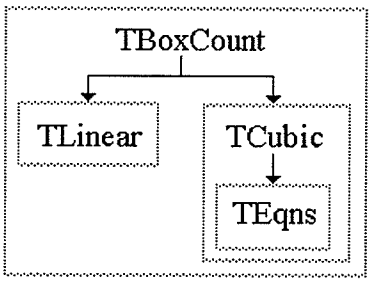


Fig. 3.12: Classes used by TBoxCount.

TRenyi0

TRenyi0 computes the Rényi and Mandelbrot dimensions for a **single scale** from a supplied

- array of points, or
- the name of a *.cdf or *.pnt file containing points

which lie on the fractal F being measured. The computed values are saved in files and not retained. Integer-only values for the Rényi exponents q are used to avoid the use of the *pow()* function and to speed up the computations. The values associated with $q = +\infty$ and $q = -\infty$ are always determined for completeness.

TRenyi

TRenyi is a slightly enhanced version of TRenyi0. The computed Mandelbrot and Rényi dimension values are saved in arrays for later recall by the calling function, and the probing scales $1/r_k$ can be changed between successive computations. This allows one instance of TRenyi to process corona discharge fractals grown on differently-sized meshes. Additionally, the minimum and maximum scales can be specified so as to eliminate outliers at the scale extremes.

TDman

TDman computes the Rényi and Mandelbrot dimensions using generalized entropy or generalized correlation methods for **multiple scales of resolution** from a supplied

- array of points, or
- the name of a *.cdf or *.pnt file containing points,

which lie on the fractal F being measured. Non-integer values of q are used in order to obtain smooth D_q and D_{MAN} curves between q=-2 and q=+2. Linear regression is used to obtain the values for D_q from measurements on multiple scales. In order to compute the derivative of $(q\text{-}1)D_q$ required by α_q, TCubic, TQuadratic, and TLinear classes are used to

fit cubic, parabolic, or linear curves to each point on the $(q\text{-}1)D_q$ function and to compute the derivatives of these curves at each point.

3.5.1.3 Hybrid classes

TGDIInfo

This class extracts basic information about the display driver settings and capabilities and prints them to a window. This is useful for troubleshooting any graphics-related problems encountered when running *Corona32.exe* on different computers. This class is a pure display class, but it generates its own data for display.

TMakeMovie

The TMakeMovie class was originally used to capture windows created by TErrorContour after every n global updates and save them to a sequence of files which could at some later time be played back in a rapid fashion by TPlayMovie in order that the propagation of corrections could be studied in detail. The frames could also be examined individually using the Windows Paintbrush© program. For a typical 510×510 mesh, each captured frame of a TErrorContour or a TField window has excessive memory (~256kb) and drawing time (~ 2 minutes) requirements. Therefore, this class is not generally used for simulations on realistic, large sized meshes.

TWatchEigenvals

This class graphs the 3 estimates of the spectral radius λ_{max}; the 1st norm estimate L_1, the 2nd norm estimate L_2, and the infinite norm estimate L_∞ (or L_C) on a single bitmap-based graph. It was an initial attempt at graphing data prior to TLGraph and TChart, and as such is not recommended for use.

TErrorWdw

This class is used to compute and display the error energy of the 3 estimates of the spectral radius; the 1st norm estimate L_1, the 2nd norm estimate L_2, and the infinite norm estimate Lc (c for Chebychev), with 1 TErrorWdw assigned to each estimate. The energy $E(t)$ of a time series $x(t)$ is computed as

$$E(t) = \int_0^\infty x^2(t)\,dt \tag{3.18}$$

The energy of the error between $x(t)$ and the computed backward average $x_{ave}(t)$ of the previous n samples of $x(t)$ is given by

$$Err(t) = \int_0^\infty \left|x(t) - x_{ave}(t)\right| dt \tag{3.19}$$

$Err(t)$ can be used to measure rapid departures of $x(t)$ from the average, indicative of numerical noise and oscillations in $x(t)$. If the calling function specified a threshold in TErrorWdw's constructor, then all values of $Err(t)$ below the threshold are drawn in blue, and all values above are drawn in red. The corresponding eigenvalue is drawn in magenta.

TErrorDist

This class computes and displays the distribution of residuals R (see Sec. 2.3.4.7.2) by counting the frequency of residuals which fall in 100 equally-spaced intervals in the interval of $0 \le R \le R_{max}$. The standard deviation σ of the distribution is calculated, as well as the percentage of residuals which are within 1 standard deviation of the maximum residual R_{max}, such that $0 \le R \le R_{max}$-σ. This provides an indication of the global relaxation activity. If most mesh points have relatively the same residual, then the majority of the residuals will be within 1 standard deviation of the maximum. This translates into active correction or relaxation of potentials occurring on a large scale, and

even if the maximum residual is less than some criterion, the majority of mesh potentials have not settled sufficiently to declare convergence. If most residuals are nearly 0, then the distribution will be skewed to the left with a few points near the maximum residual. This leads to a very small percentage of residuals within 1 standard deviation of the maximum residual. The interpretation of this is that the potentials of the majority of mesh points have settled very close to their final values. This provides a statistically-based, self-adjusting measure of convergence.

3.5.1.4 Dialog classes

Dialog classes are used to request filenames and parameters. As there are numerous dialog classes used by the three main programs, they will not be discussed here. Their purpose and nature should be self-evident from the calling context, from the supplied in-line documentation, and from their simple coding.

3.5.1.5 Extended options manipulation classes

TConfig

TConfig is used to manage the configuration parameters for

- the simulation topology,
- specific dielectric breakdown model parameters,
- the random generator to be used,
- the local and global solution algorithms, and
- the convergence criteria.

TConfig is responsible for

- sequencing the dialog boxes to capture and check the user preferences when creating a new simulation file,
- storing these parameters to a file, and
- checking these parameters when reading them from a file.

3.5.1.6 Composite classes

TConvergence

TConvergence is used to determine the convergence of the iterative solution method. Many criteria are involved, and the convergence criteria are specified by the user via TConfig when the simulation is first created. TConvergence has more convergence criteria than TAccelSolver. The class structure is shown in Fig. 3.13.

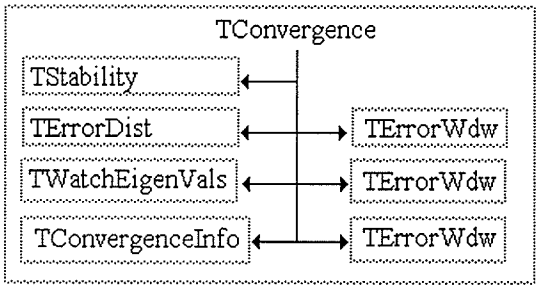


Fig. 3.13: Relationship of classes in TConvergence.

The calling class must initialize TConvergence prior to any iterations passes to set the iteration method, the maximum iterations (or bailout level), the initial acceleration factor if applicable, and the pen colour, used by some display classes. After a specified number of global iterations, the IsDone member function is called to determine the state of convergence. If needed, it first computes the 1^{st}, 2^{nd}, and infinite norms (Sec. 2.3.4.7.3) of the mesh residuals (Sec. 2.3.4.7.2). The general sequence of events is shown in Fig. 3.14.

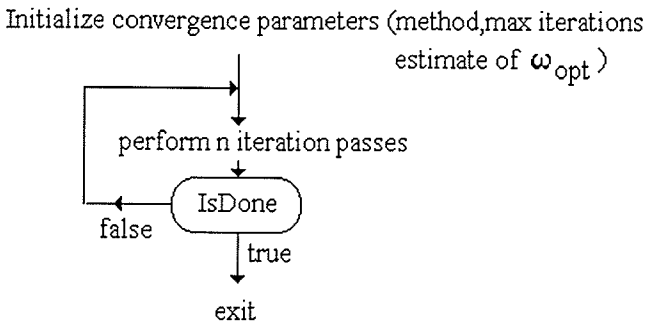


Fig. 3.14: Role of TConvergence in an iterative method.

The convergence tests are described below. The complete C++ code listings can be found in Appendix H [Aren96].

1. Test for end of iterations

$$Iteration\,Count > Maximum\,Count \tag{3.20}$$

This test is done first. If we have reached the maximum number of iterations passes allowable, then obviously convergence has not been attained on previous pass, so exit at this point without checking any other criteria. The inclusion of a *Maximum_Count* allows the iteration process to *bail out* of infinite loops caused by unattainable or unrealistic convergence criteria.

2. Tests on norms

$$\| r \|_{\infty} = 0 \tag{3.21}$$

If the infinite norm is 0, then all the residuals have vanished, and convergence has been attained. This condition also implies that the 1st and 2nd norms have also vanished. If this condition is true, the other norm tests are irrelevant.

$$\| r \|_\infty > \textit{Maximum Potential} \tag{3.22}$$

If the maximum residual is greater than the maximum potential, then major oscillations of mesh potentials are occurring. The current iterations should be halted, and the acceleration factor should be corrected. This condition can only occur for *accelerated* iterative methods.

$$\| r \|_\infty < \textit{acceptable worst case residual error} \tag{3.23}$$

Convergence can be declared if the worst case residual error, which happens to be the infinite or maximum norm of the residual, is less than the maximum acceptable residual

$$\frac{\sum_{1}^{N} \Theta\left(r_j - \left(\|r\|_\infty - \sigma\right)\right)}{N} \leq \textit{specified percentage} \tag{3.24}$$

This test measures the percentage of the residuals which are within 1 standard deviation of the maximum residual (see TErrorDist in Sec. 3.5.1.3), measuring the relaxation activity still occurring. This test is optional.

3. Compute spectral radius estimates

Eq. 2.76 is used to estimate the spectral radius λ_{max}. λ_1 is the estimate of λ_{max} derived from the 1st norm (Eq. 2.72), λ_2 is the estimate derived from the 2nd norm (Eq. 2.73), and λ_C is the estimate derived from the infinite norm (Eq. 2.74) of the residuals.

4. Tests on spectral radius estimates

$$\lambda_1 \geq 1.0, \lambda_2 \geq 1.0, \lambda_\infty \geq 1.0 \tag{3.25}$$

If any estimated spectral radius exceeds 1.0, then some of the eigenvalues associated with the equivalent iteration matrix are complex, and at least some mesh potentials will oscillate. If the spectral radius estimate which is used to estimate the acceleration factor is exactly 1.0, then the estimated acceleration factor for SOR and SLOR will be ω_{opt} = 2.0 resulting in divergence instead of convergence. For some simulations, a point is reached where one or more norms no longer change, indicating that the global relaxation process has been arrested, causing the associated spectral radius estimate(s) to equal 1.0. Each test is individually enabled.

$$\lambda_X = \lambda_{Xaverage} \pm k_S\%, \quad x = 1,2,\infty \tag{3.26}$$

This test measures the stability of the spectral radius estimates. If the spectral radius estimate(s) are sufficiently stable for a specified number of iterations I_S, where I_S is large and k_S is small, then the initial transients can be considered to be extinguished. This condition does not imply *convergence*, but that the process *is converging*. The test for each estimate is individually enabled.

$$\lambda_X > \lambda_{Xaverage} + k_{INCR}\%, \quad x = 1,2,\infty \tag{3.27}$$

This test terminates the iteration process if the estimates of the spectral radius suddenly increase by a specified amount k_{INCR} above the running average, usually indicative of some oscillations of mesh potentials. In order for both Eqns. 3.26 and 3.27 to be used, $k_{INCR} > k_S$. The test for each estimate is individually enabled.

$$Err(\lambda_X) > threshold_X, \quad x = 1,2,\infty \tag{3.28}$$

This test was designed to detect accumulated error in the spectral radius estimates.

$$D_\sigma(\lambda_X) < 2.0, \quad x = 1,2,\infty \tag{3.29}$$

This test attempted to apply the variance dimension to the spectral radius estimates to detect the catastrophic introduction of numerical noise due to truncation and rounding. Further study of the application of this method is recommended in order to properly set the window size, window shift, and probing increment parameters.

5. Compute acceleration

   The estimate of the spectral radius is used to estimate the optimal acceleration factor.

6. Tests on estimate of optimal acceleration factor

$$\omega_{est} > \omega_{opt} \tag{3.30}$$

   This test checks if the estimated acceleration factor exceeds the optimal acceleration factor for the mesh in the absence of all singularities. Since the norms are estimates, it is possible to over-estimate the acceleration factor even if the spectral radius is less than 1.0. This test just corrects ω_{est}, it does not make any statement about the state of convergence.

Experiments show that the accuracy of estimate of the spectral radius improves with the number of iterations (Eq. 2.77). For simulations which employ a few global iterations (such as 8 global passes in TStacey),

- the estimates of the spectral radius will not be very accurate, therefore any convergence criterion based on the analysis of the spectral radius estimates must be

used with extreme care and caution, and,

- tests based on Eqns. 3.21 through 3.23 should not be utilized due to the initial instability of the norms.

One improvement that could be made to TConverge is the use of only pure computational classes to determine the state of convergence, with the parent of TConverge responsible for managing the display classes. This would permit the use of TConverge with non-graphical versions of the software running on non-Windows operating systems.

TVarWdw

TVarWdw is used to display the variance dimension D_σ of the 3 estimates of the spectral radius λ_{max}; the 1st norm estimate L_1, the 2nd norm estimate L_2, and the infinite norm estimate Lc. The calling function passes the estimates to TVarWdw which routes each estimate to its own TVariance class. The computed dimensions are then routed to separate TChart classes for display. This allows the estimates to be visually monitored and for the computed dimensions to be used by the TConvergence class. The classes employed by TVarWdw are shown in Fig. 3.15.

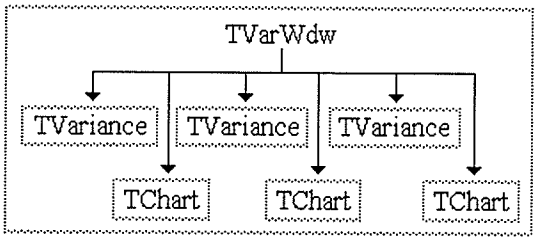


Fig. 3.15: Relationship of classes in TVarWdw.

TConvergenceInfo

This class utilizes 2 TChart classes to display information about the convergence of global updates: one to display the convergence codes, and the other to display the number of iterations before convergence was determined or the maximum number of passes were completed.

TBoxCountDim

This class computes and displays the Hausdorff mesh counting dimension, utilizing (see Fig. 3.16)

- a TBoxCount class to compute the $\log(N_k)$ and $\log(1/r_k)$ data,
- a TChart class to plot the trajectory of
    - ◊ the linear regression slope from TLinear using blue "×"'s, and
    - ◊ the maximum slope of a fitted cubic polynomial using magenta "×"'s, and
- a TLGraph to plot the $\log(N_k)$ and $\log(1/r_k)$ data for the Hausdorff mesh-counting dimension D_{HM} over time.

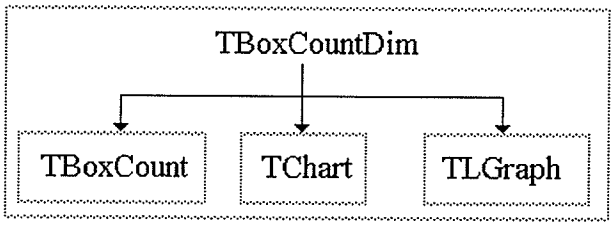


Fig. 3.16: Relationship of classes in TBoxCountDim.

The calling function can specify linear or dyadic intervals useful for showing the effects of the growth of the Laplacian fractal on the dynamics of the dimension computations. The window can be saved to the clipboard.

TSpectrum

This class computes and displays the Rényi and Mandelbrot dimensions, utilizing

- a TRenyi class to compute the Rényi and Mandelbrot dimensions
- a TLGraph class to plot the Rényi dimension curve[s], and
- a TLGraph class to plot the Mandelbrot dimension curve[s].

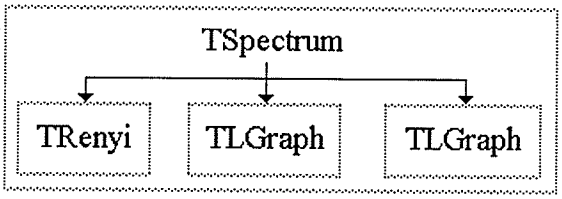


Fig. 3.17: Relationship of classes in TSpectrum.

The calling function can specify the sampling interval size, interval type {linear or dyadic} the range of generalized integer exponents q, and the basic method: generalized entropy or generalized correlation. TSpectrum can be used to show the effect of growth on the dimensions, or to compare the dimensions of different Laplacian fractals. The TSpectrum window can be saved to the clipboard.

3.5.2 Functions

The functions provided by *Corona32.exe* can be categorized into 3 main categories: functions which allow the effects of modelling and solution methods of the original implementation [Stac95] to be observed, functions which implement a general purpose simulation engine, and functions which permit the multifractal analysis of the resultant discharges.

3.5.2.1 Study of Effects of Modelling and Solution Methods

Several menu options permit the study of the effect of:

- the selection of a random number generator, or
- the selection of the iterative method on the resultant Laplacian fractals,
- the effect of the estimation of the optimal acceleration factor ω_{est} based on the estimate of the spectral radius λ_{max}
    - ◊ computed by classical methods from the 1st norm of the residuals, and
    - ◊ computed from the largest eigenvalue of the singularity-free regions,
- the effect of dynamic global and/or local acceleration based on either of the above methods for estimating λ_{max} on the first and infinite residual norms for point, line and block SOR methods.

The functionality for these options are provided by the TStacey, TNewWest, and TAccTrees classes which are extensions of the implementation developed by Stacey [Stac95] and Arendt [Aren94], with the provisions to experiment with slightly different parameters.

3.5.2.2 Generalized Simulation Engine

A generalized simulation engine allows simulation topologies, solution and convergence parameters to be defined and stored in corona definition format (*.cdf) files via a TCreate class. Topologies other than parallel plates, such as coaxial and square outer electrodes are supported. Creation of batch files via TBatch allows multiple simulations to be run sequentially, unattended. Each simulation is controlled by a TCase class which creates windows to monitor:

- the spatial distribution of the residuals (TErrorContour),
- the statistical distribution of the residual (TErrorDist),
- the potential field (TFieldView),
- the location of all candidate growth sites and all mesh sites at 0 potential (TGrowth),
- the Hausdorff mesh counting dimension trajectory versus a dyadic growth stage (TBoxCountDim),
- the trajectories of the generalized entropy and correlation implementations of the Rényi and Mandelbrot dimensions (TSpectrum),
- the trajectories of the variance dimension (D_σ) of each of the residual norms,
- the error energy of the spectral radius estimates based on the 3 norms (TEnergy), and
- the cause of global convergence for each growth stage (TConvergenceInfo).

3.5.2.3 Measurement of Fractality and Multifractality

Several options allow the fractality and multifractality of the generated Laplacian fractals as exhibited by

- the D_q vrs. q and f_α vrs. α_q curves of the Rényi and Mandelbrot dimensions based on
    - ◊ the generalized entropy, and
    - ◊ the generalized correlation methods,
- the Hausdorff mesh counting dimension (D_{HM}) implementation of the self-similarity measure,

to be computed for any number of discharges based on measurements from a single or multiple scales, showing just the final dimension characteristics or the trajectory of the dimension as the discharge grows.

CHAPTER 4

EXPERIMENTAL RESULTS AND DISCUSSION

4.1. Introduction

Several experiments were designed to evaluate the suitability, performance, and impact of various dielectric breakdown model parameters and iterative methods of solving for the mesh potentials. For each experiment, where applicable,

- the purpose,
- the implementation,
- the results, and
- the analysis and impact

of the experiment are described.

4.2. Experiment 1: Effects of Model Parameter Variations

The dielectric breakdown model employed by Stacey [Stac95] was used as the reference model to evaluate the effects of mesh potential representation, random generator, and iterative methods, and other parameters. It should be restated here that mesh points having potentials of 0 (the minimum electrode and the corona potential) or *MaxV* (the maximum electrode potential) are treated as Dirichlet boundaries, and are not updated by any iterative methods. The discussion here applies only to parallel plate topologies.

4.2.1 Correction of Mesh Initialization

Assuming a mesh having szy=5 rows of points, an electrode along the top row having a maximum potential of $MaxV = 400$, an electrode of 0 potential along the bottom row, then Stacey's initialization formula of

$$\phi_{row} = MaxV\left(\frac{szy-1-row}{szy}\right) = 80(4-row), \quad row = 0,1,2,3,4 \tag{4.1}$$

yields row potentials of ϕ_{row} = {320, 240, 160, 80, 0}. After noticing that the actual potential of top row was not $MaxV$, the following corrected initialization formula was used

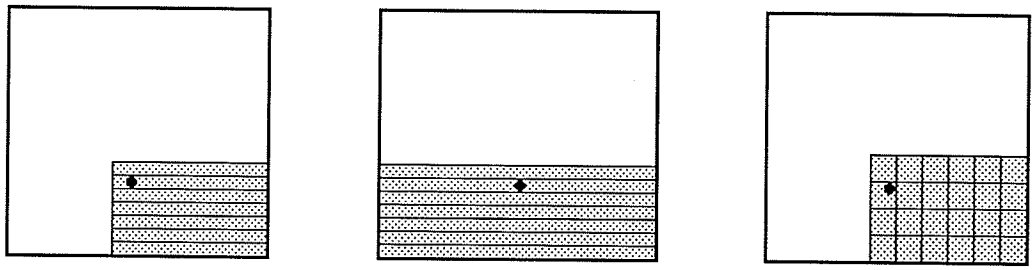
$$\phi_{row} = MaxV\left(\frac{szy-1-row}{szy-1}\right) = 100(4-row), \quad row = 0,1,2,3,4 \tag{4.2}$$

which yields the correct row potentials of ϕ_{row} = {400, 300, 200, 100, 0}. Raising the mesh potentials has the effect of increasing the growth potentials, affecting the growth of the discharge, so the discharges generated "exactly" as per Stacey's model were not exact replicas, but closely resembled his published results [Stac95].

4.2.2 Effect of Iteration Mesh Scanning

Standard mesh scanning direction for the updating of the mesh potentials is from left to right (point and block methods), and from top to bottom (all methods). This convention is determined from the fact that the equivalent iteration matrices require an ordered, non-random, updating of the unknown mesh potentials. Altering the scanning order can change the eigenvalues, and hence the spectral radius, affecting the convergence rate and possibly resulting in non-convergence.

If an $N \times N$ mesh experiences a disturbance at a point •, and assuming conventional mesh scanning, then the pattern of "downwind" points which are affected as a result on the next global rescan is shown (Fig. 4.1) for point, line and block iterative methods.



a. point methods b. line methods c. block methods

Fig. 4.1: Effect of normal scanning on propagation of disturbance at •.

In Stacey's parallel plate model, the corona grows from the bottom plate upwards, drastically altering the field at the latest growth site. However, the mesh rows are updated in the standard top to bottom fashion. This means that the effect of the corona growth related disturbance on row r will always be propagated to the bottom plate each global iteration pass, but will only reach the top plate after

$$t_{r=0} = \frac{r-1}{8} \tag{4.3}$$

global iterations. So for a 510×510 mesh experiencing a disturbance on row 508, it takes 507 global iteration sweeps, equivalent to 64 more corona growth epochs having 8 global sweeps per growth epoch for the effects of the original disturbance to reach the topmost mesh row. If the mesh scanning direction was reversed, then the effects of the disturbance would always be propagated in the general direction of the corona growth, but depriving the lower rows from experiencing the immediate effects. To assess the impact of the iteration scanning direction, two simulations (**Trirn** and **Trfrn**) employing the normal and reverse row-order scanning direction were run (Sec. 4.2.6, App. E).

4.2.3 Effect of Tag Array Scanning

Since the potentials of points close to the actual corona discharge can be reduced to 0 due to screening effects when using integer representation, Stacey employed a tag array to record the actual corona points so that the discharge fractal could be extracted from the midst of all the 0 potential points. When computing the growth probabilities, his implementation scanned the tag array, left to right, top to bottom, searching for candidate growth points. To assess the impact of this scanning direction, runs were performed in which the tag array was scanned left to right, but bottom to top (Sec, 4.2.6, App. E).

4.2.4 Effect of Potential Representation

In the DBM employed by Stacey [Stac95], non-accelerated Jacobi and Gauss-Seidel methods were employed, and 32-bit integers were used to represent the mesh potentials. Since accelerated methods utilize non-integer acceleration factors such that

$$1.0 \leq \omega < 2.0 \tag{4.4}$$

floating-point numbers are used to represent the mesh potentials so as to avoid the overhead required to convert integers to floating-point for computation of the updated potentials and then to convert back to integer format for storage. Runs were performed to assess the effect of mesh potential representation (Sec. 4.2.6).

4.2.5 Undershoots and Overshoots

When using accelerated methods, undershoots (potentials less than 0) and overshoots (potentials greater than *MaxV*), both which are artifacts of the iterative solution approach, can occur. Undershoots are clipped to 0.0, otherwise negative or complex growth probabilities could result from Eq. 2.12, as illustrated by these examples.

Example 4.1: $\phi_{x,y} = -1, \quad \eta = 1.0, \quad \text{then} \quad P(x,y) = \dfrac{-1}{\sum_{i,k} \phi_{i,k}^{\eta}} < 0$

Example 4.2: $\phi_{x,y} = -1, \quad \eta = 1.5, \quad \text{then} \quad P(x,y) = \dfrac{-1^{1.5}}{\sum_{i,k} \phi_{i,k}^{\eta}} = \dfrac{-i}{\sum_{i,k} \phi_{i,k}^{\eta}}$

Example 4.3: $\phi_{x,y} = -1, \quad \eta = 1.65, \quad \text{then} \quad P(x,y) = \dfrac{-1^{1.65}}{\sum_{i,k} \phi_{i,k}^{\eta}} = \dfrac{0.454 - i0.89}{\sum_{i,k} \phi_{i,k}^{\eta}}$

Undershoots tend to be small since they occur near the screened regions. Clipping them to 0, which introduces a fixed boundary at those points, has not resulted in side effects, except for possibly the accelerated SBOR run. Overshoots tend to be reduced to below *MaxV* on the next iteration pass, so they are not clipped to *MaxV* nor adjusted.

4.2.6 Discussion

Table 4.1 summarizes the dimensions of various runs performed with changes in a single parameter. The corresponding Mandelbrot multifractal curves appear in Sec. 4.9. Examining the computed morphological dimensions, it can be seen that the multifractality of Laplacian fractals is relatively insensitive to these varied parameters. If we want to split hairs, then we can observe that using floats instead of integers for mesh potentials generally results in *minutely* higher dimensions. This is most likely due to the fact that integer potentials in the screened regions will be truncated to 0, whereas floating point potentials become fractional ($0.0 \leq$ potential ≤ 1.0), retaining a growth probability however small, and thereby resulting in a slightly denser fractal object.

For the runs in Table 4.1, the ranges for the morphological dimensions are:

- $D_0 = 1.6750 \pm 0.67\ \%$
- $D_1 = 1.6680 \pm 0.64\ \%$
- $D_2 = 1.6630 \pm 0.69\ \%$

Thus, dimensions are affected by less than 1% by the described parameter variations. The above range of D_2 intersects the range of D_2 computed by Stacey [Stac95] which was $1.6764 \pm 0.90\ \%$.

Table 4.1: Effect of model implementation on dimensions

	Parameters			Dimensions		
file name	potentials	iteration scanning	tag array scanning	D_0	D_1	D_2
trinn	integer	normal	normal	1.6741	1.6690	1.6654
trfnn4	floats	normal	normal	1.6845	1.6773	1.6722
trinr	integer	normal	reverse	1.6639	1.6574	1.6515
trfnr4	floats	normal	reverse	1.6651	1.6573	1.6519
trirn	integer	reverse	normal	1.6798	1.6727	1.6679
trfrn4	floats	reverse	normal	1.6828	1.6746	1.6691

4.3. Experiment 2: Timing of Execution of Methods

The second evaluation of the iterative methods consisted of an experiment to determine the required execution times t for each method, so that they could be ranked

according to speed. Additionally, the power law relationship between t and the mesh width N could be checked. This experiment was performed using a 486DX-33 based computer.

4.3.1 Mesh with no Interior Singularities

To perform the experiment, 125×125, 250×250 and 500×500 meshes with Dirichlet boundaries along the mesh edges were employed (see Fig. 4.2). The interior points were initialized to a constant value. The times required (to the nearest 0.01 sec.) by each method to perform 100 global sweeps were recorded and are shown in Figs. 4.4. A graph of t versus N for all methods appears in Fig. 4.5.

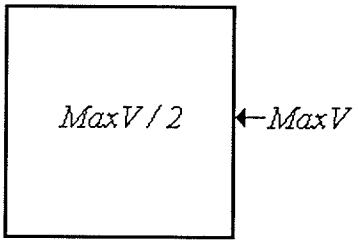


Fig. 4.2: Mesh initialization for experiment 2.

4.3.2 Mesh with 9.83% Interior Singularities

From Stacey's results and from preliminary experiments, it was estimated that the ratio of corona points to interior mesh points had an upper bound of about 10%. To ascertain if the introduction of singularities into the mesh interior due to the corona had any effect on the solution times, experiment 2 was also performed with a random seeding of 0 potential singularities at about 9.83% of the interior mesh points. Figure 4.3 shows the timing difference due to the introduction of singularities.

4.3.3 Discussion

From Fig. 4.4, it can be seen that non-accelerated methods execute faster than their accelerated counterparts and derivatives. This arises from the additional complexity required by the accelerated methods, such as the multiplication of the residual by ω and the addition of the $U_{i,j}(1-\omega)$ term (App. C and D). It should be obvious that while accelerated methods accelerate the convergence rate, the execution time is not. However, considering only accelerated methods which are our primary interest, the ranking of methods from fastest to slowest appears in Table 4.2.

Table 4.2: Relative Execution Speed Ranking.

Iterative Method	Execution Ranking
SOR	fastest
SBOR	↑
SORBW	
SBORBW	
SLOR	execution speed
SLORBW	
ADSOR	
ADSBOR	
ADSLOR	↓
ADSLORBW	slowest

For any method m applied to a square mesh of width w, the execution time $t_{m,w}$ scales with the square of the ratio a of the widths w_2 and w_1, such that

$$a = \frac{w_2}{w_1}, \quad \frac{t_{m,w_2}}{t_{m,w_1}} = k^2 a^2, \quad \text{where} \quad 1.0237 \le k \le 1.045 \tag{4.5}$$

which is within 5% of the ideal of k=1.00. The introduction of singularities at 9.83% of the interior points caused a maximum reduction of the execution time reduction of 11.8% for SOR and a minimum of 6.77% for SBOR, neither of which will have a significant

impact on the overall simulation time. SBOR has significant overhead to determine and access the proper stencil, hence its lower reduction.

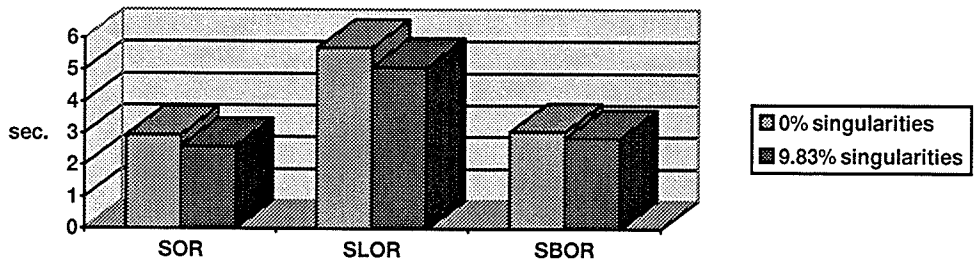
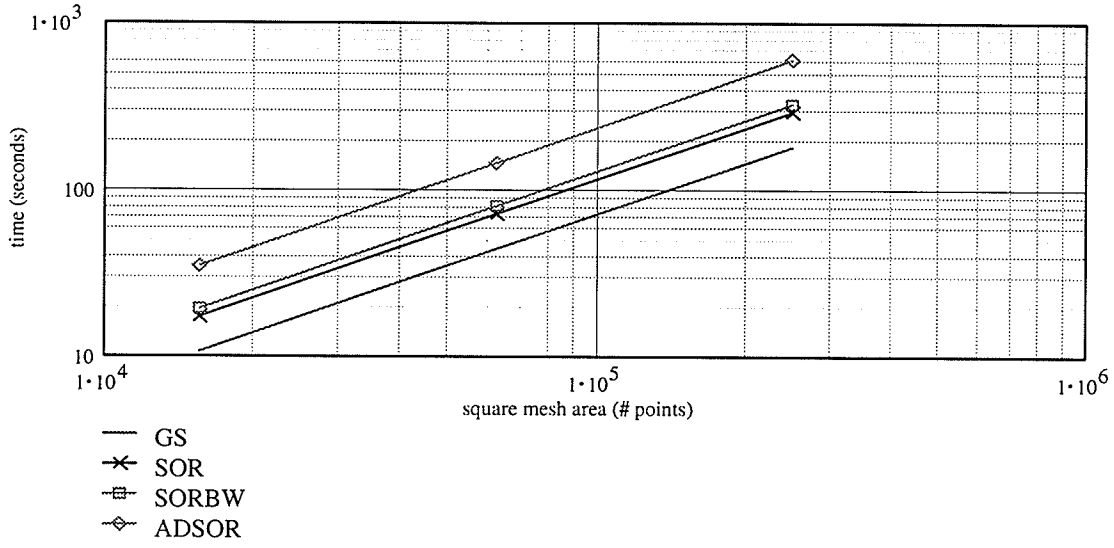
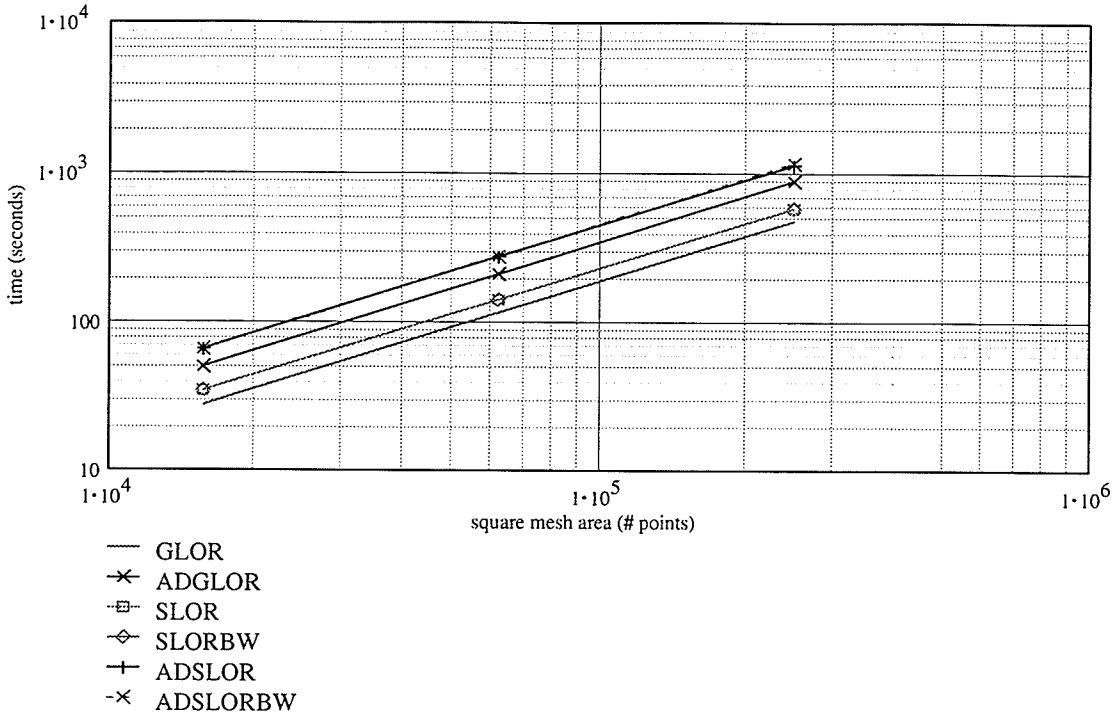


Fig. 4.3: Effect of singularities on solution times.

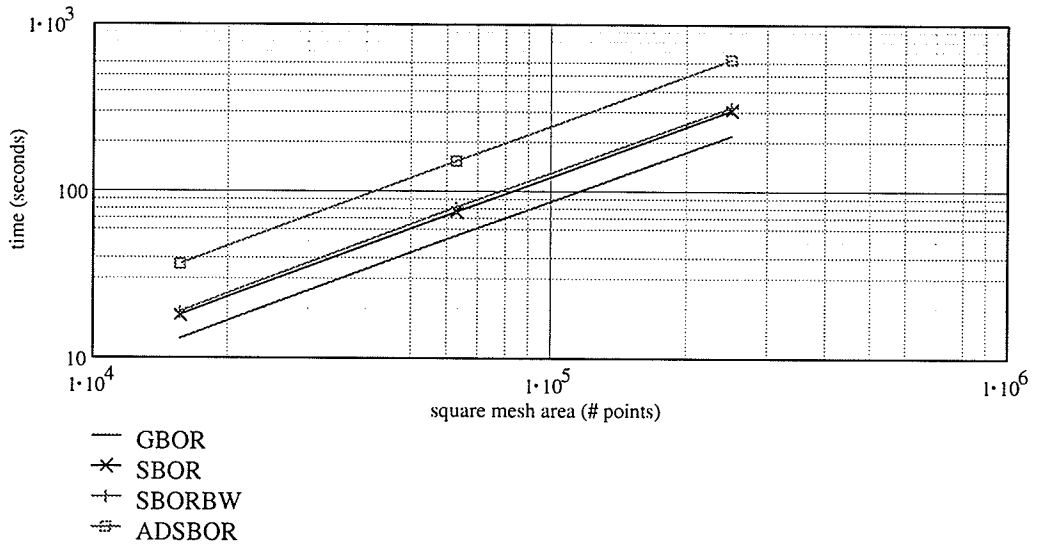


a. Execution times for point methods

Fig. 4.4: Execution times for Point, Line, and Block Methods.



b. Execution times for line methods



c. Execution times for block methods

Fig. 4.4: : Execution times for Point, Line, and Block Methods (cont'd).

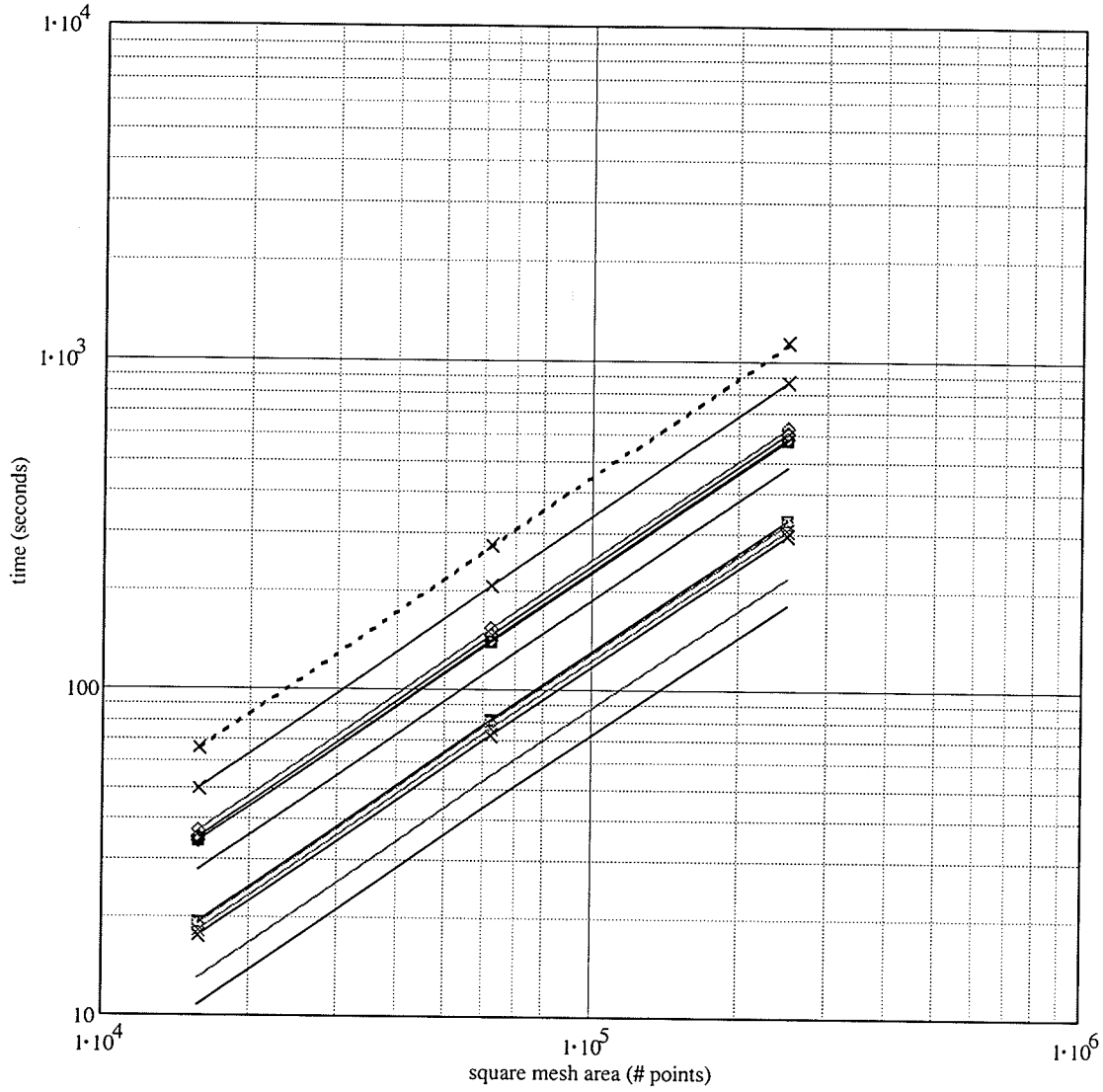


Fig. 4.5: Comparison of Execution Times For All Methods.

4.4. Experiment 3: Iterations Until Convergence

The second evaluation of the iterative methods consisted of an experiment to determine the optimal acceleration factor ω_{opt} and the corresponding optimal number of iterations N_{opt} for various square meshes and other rectangular meshes.

4.4.1 Experimental Setup

The test configurations consisted of 8×8, 16×16, 32×32, 64×64, and 128×128 mesh sizes with Dirichlet boundaries as shown in Fig. 4.6. The potentials of the interior points were set to 1.0.

The number of iterations N, upto a maximum of 2000, corresponding to 21 fixed values of ω in the range of $1.0 \leq \omega \leq 1.99999$ were determined and saved in a list sorted according to ω. The convergence criterion was a maximum residual error of less than 100, equivalent to 0.01% of *MaxV*. The range of ω was upper bounded by the first ω for which numerical oscillations were detected, an indication of over-acceleration ($\omega > \omega_{opt}$). The software then attempted to home in on ω_{opt} by scanning the list for the minimum value of N and repeating the simulation using the values of ω (ω_L and ω_R in Fig. 4.7) midway between $\omega(N_{min})$ and the 2 adjacent values of ω. The new (ω_L, N) and (ω_R, N) pairs were inserted into the list, and the homing process was terminated after 16 points were added to the list in this fashion. This experiment was performed using a Pentium-150 based computer to reduce the run time.

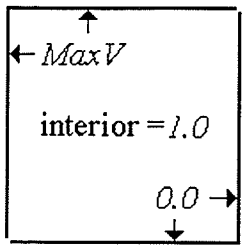


Fig. 4.6: Mesh initialization for experiment 3.

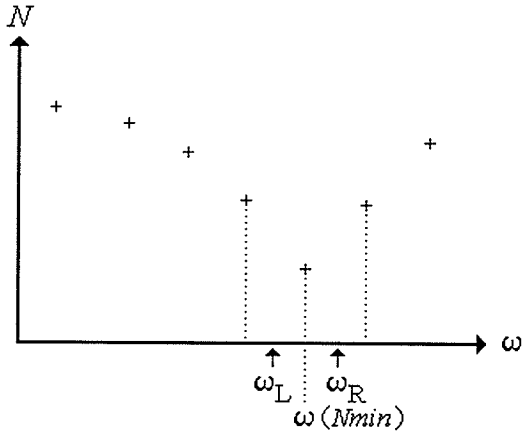
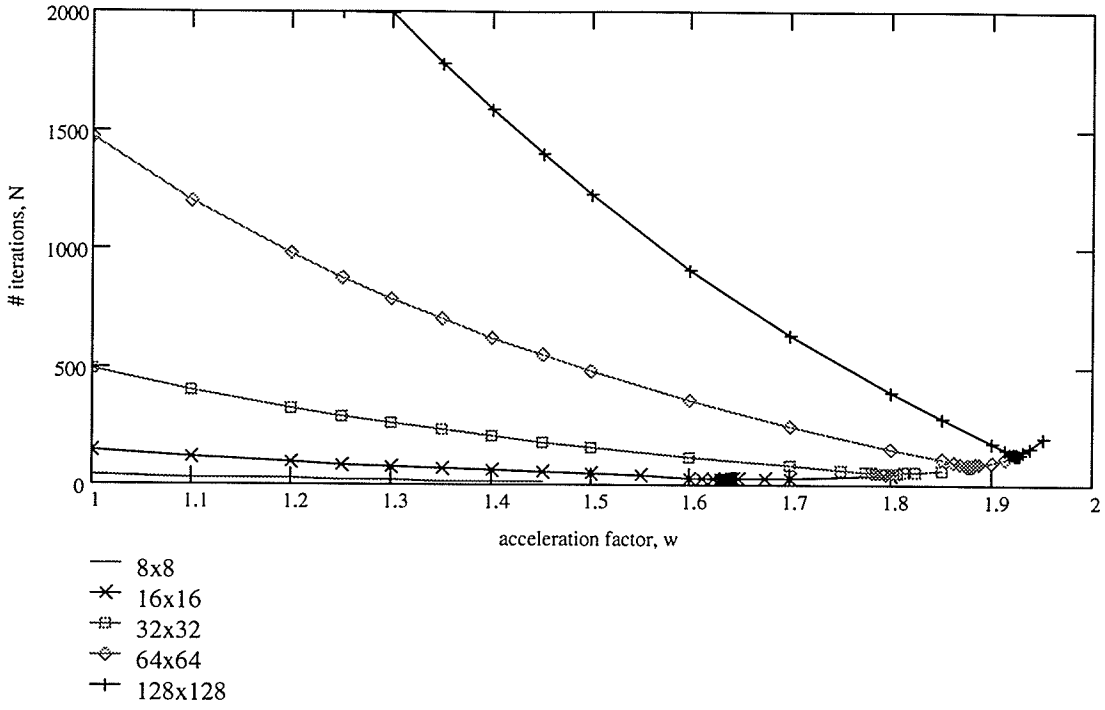


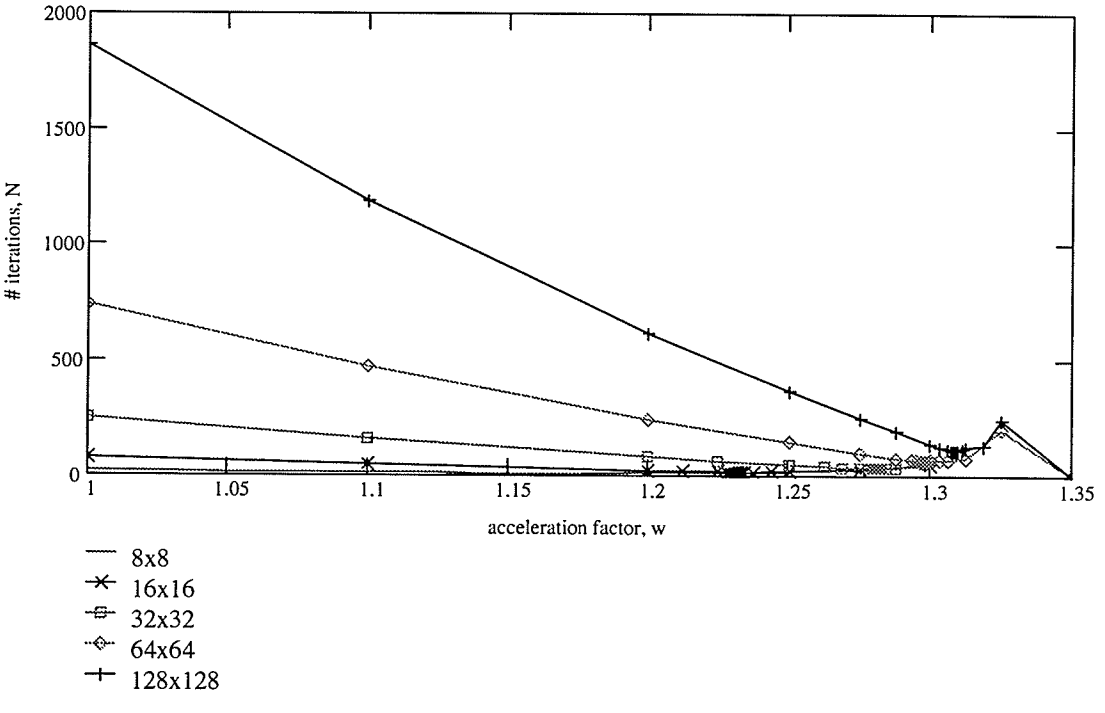
Fig. 4.7: Homing in on ω_{opt}.

4.4.2 Experimental Results

Figure 4.8 shows the N versus ω curves for SOR, SLOR, and SBOR iterative methods as a function of square mesh size. Figures 4.9 to 4.11 show the N versus ω curves for point line and block iterative methods for a square mesh of width w=64.

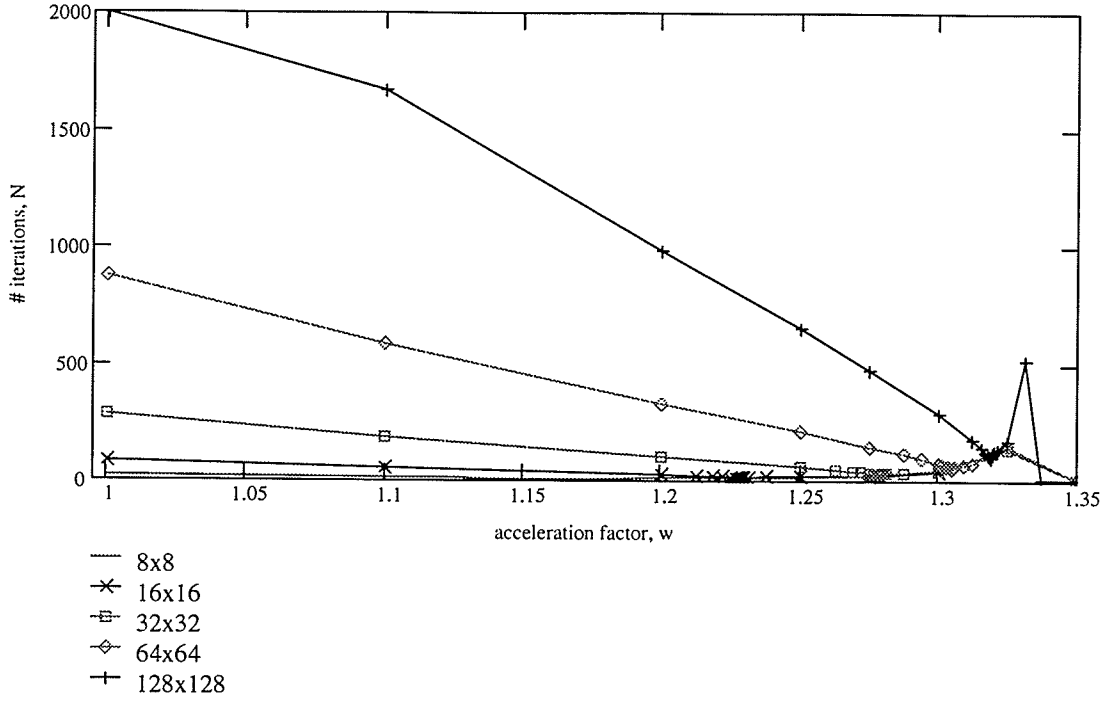


a. SOR



b. SLOR

Fig. 4.8: N vrs ω for various mesh sizes.



c. SBOR

Fig. 4.8: *N* vrs ω for various mesh sizes (cont'd).

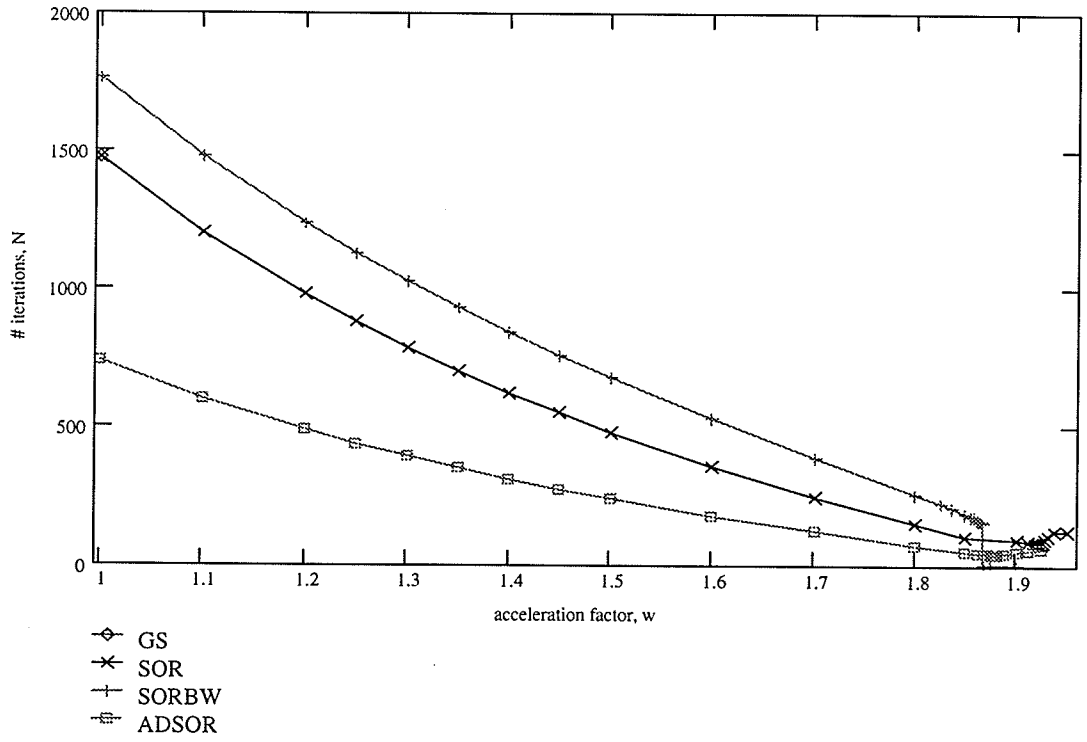


Fig. 4.9: *N* vrs ω for various point methods.

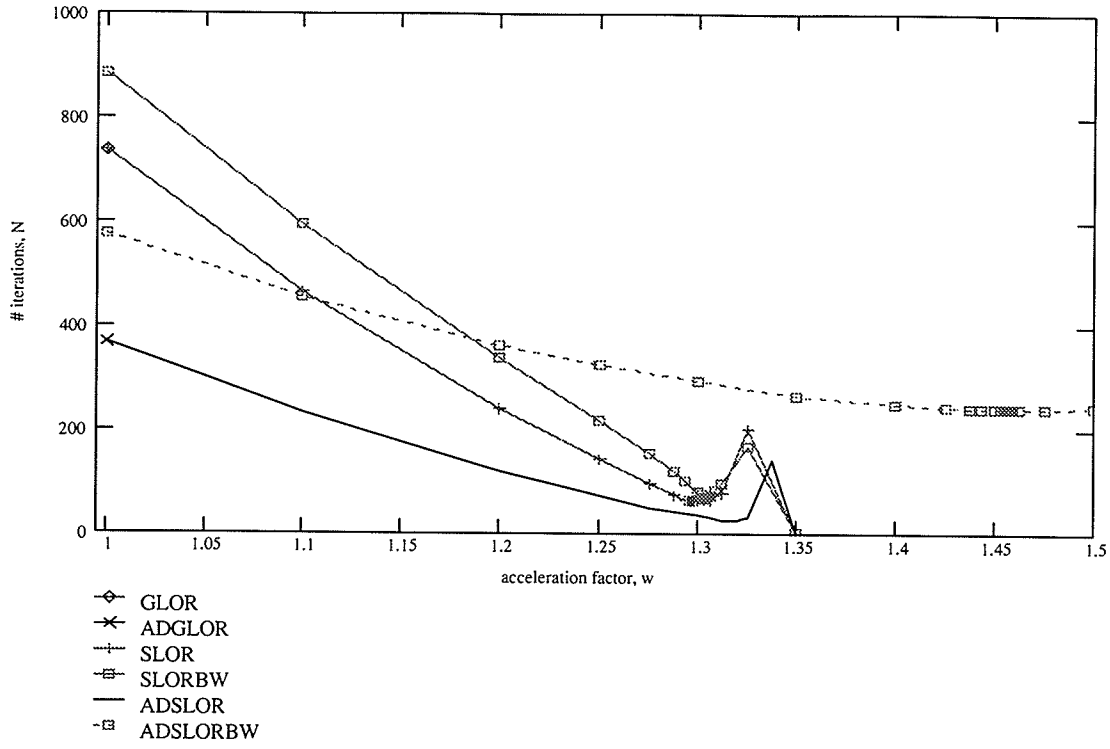


Fig. 4.10: N vrs ω for various line methods.

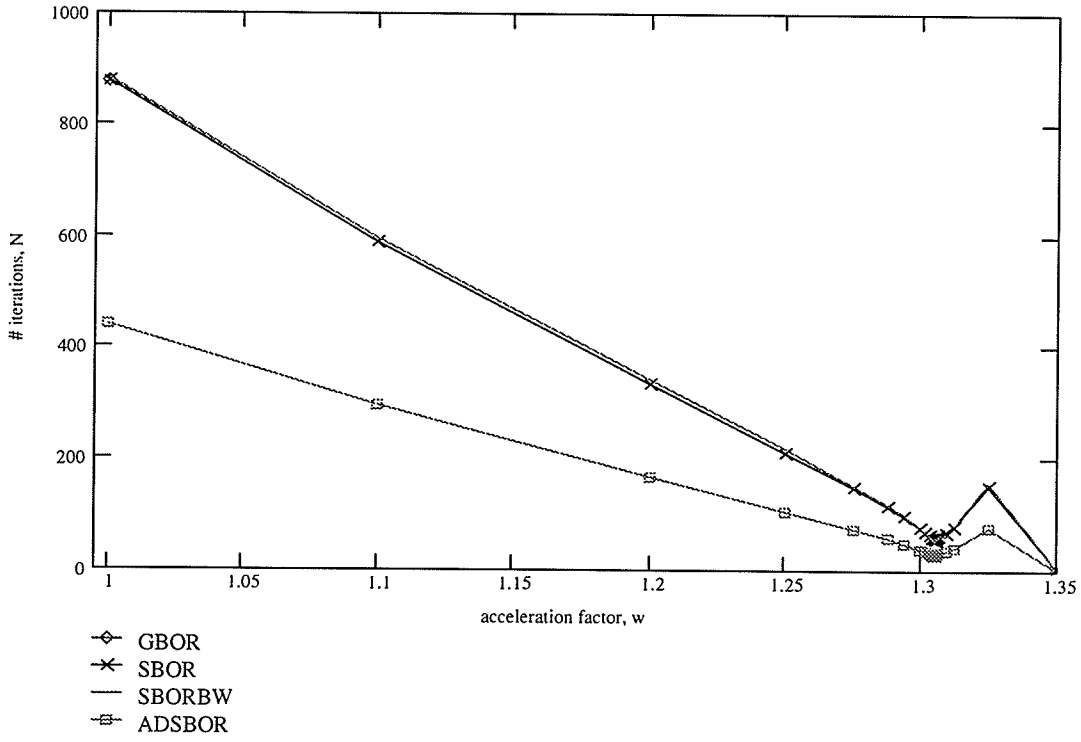


Fig. 4.11: N vrs ω for various block methods.

4.4.2.1 Comparison with Theoretical Values

Upon examining the output data file containing the N and ω data, it was noticed that in most cases, the minimum value of N was not characterized by a sharp dip at a single value of ω, but by a flat region. In these cases, a region containing the *unknown exact value* of (N_{opt}, ω_{opt}) was bounded by the first values of $N>N_{opt}$ on either side as shown in Fig. 4.12. In general, the greater the mesh size, the smaller the bounding interval and the better the resolution of the determined ω_{opt}.

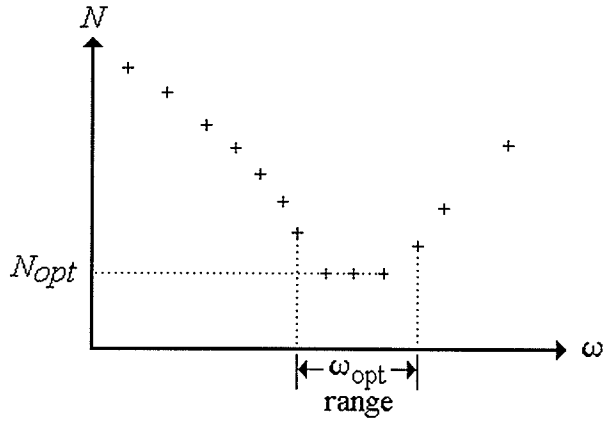


Fig. 4.12: Bounding the range of ω_{opt} from experimental results.

Figures 4.13, 4.14 and 4.15 contrast the experimental ranges and the exact theoretical values of ω_{opt} for 8×8, 16×16, 32×32, 64×64, and 128×128 mesh sizes for the SOR, SLOR and ADSLOR methods respectively. The experimental ω_{opt} for SOR is within 3% of the predicted values. As the mesh size increases, the formulas tend to overestimate the experimental ω_{opt} for SOR and SLOR, and slightly underestimate the experimental ω_{opt} for ADSLOR. Figure 4.14 indicates that classical formula published by Ames, Young, and others is grossly incorrect, and that Evans' formula for SLOR is much more accurate.

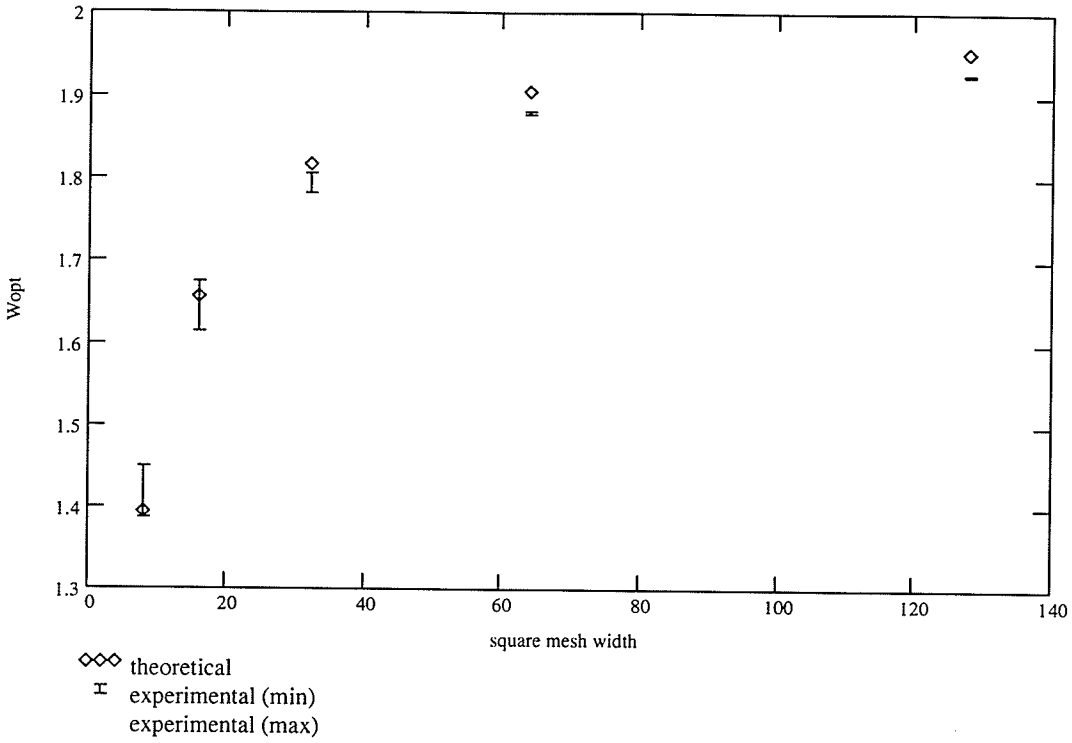


Fig. 4.13: ω_{opt} for SOR vrs square mesh width.

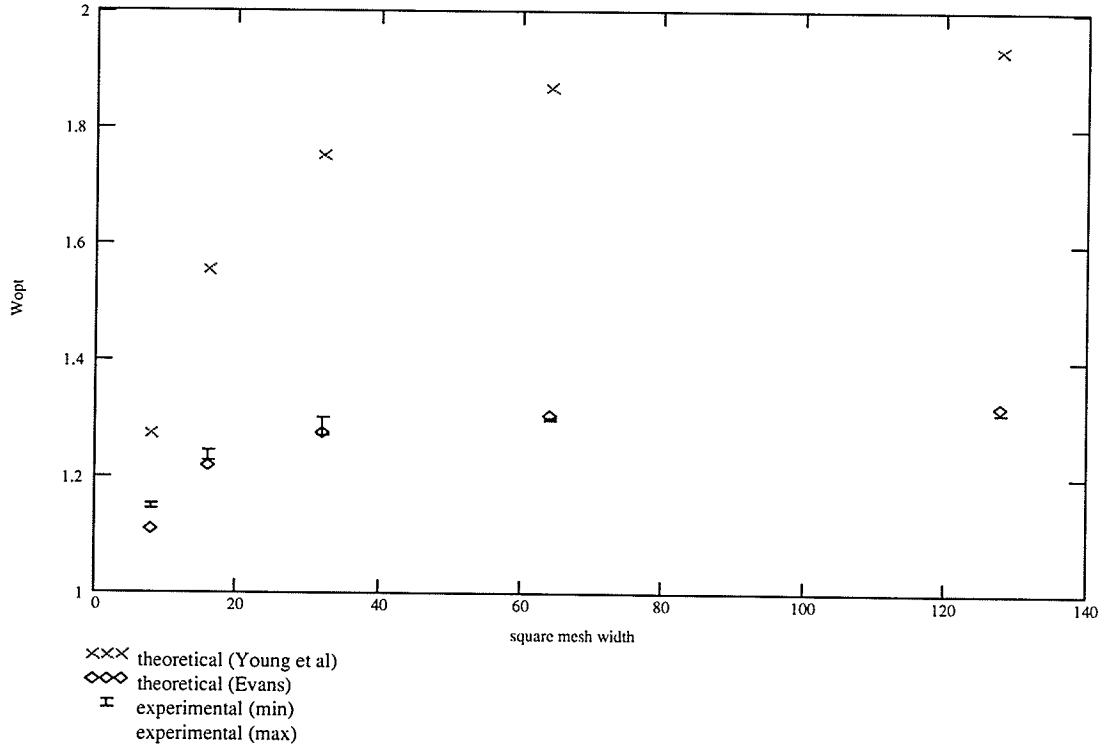


Fig. 4.14: ω_{opt} for SLOR vrs square mesh width.

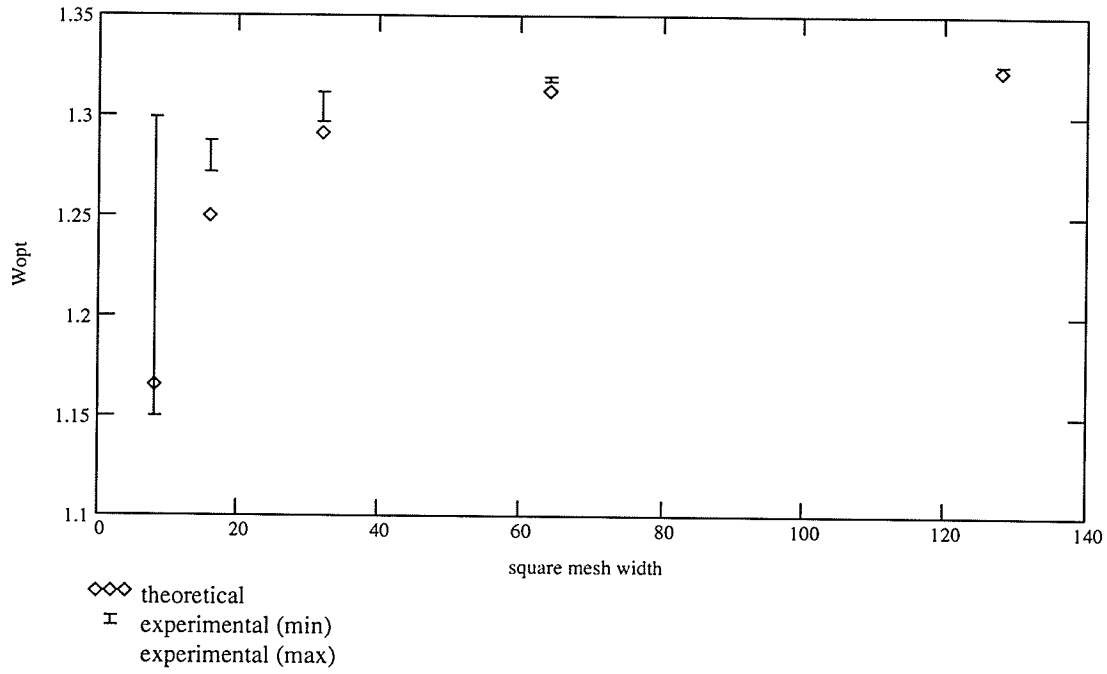


Fig. 4.15: ω_{opt} for ADSLOR vrs square mesh width.

4.4.2.2 Implications for SBOR

No published formulas for λ_{max} or ω_{opt} for SBOR were uncovered during the literature search for this thesis, however, by examining the experimental results in Fig. 4.16, it can be seen that for *square meshes* containing $n \times n$ points,

$$\omega_{opt}(SBOR) \geq \omega_{opt}(SLOR) = \frac{4}{3\left(1 + \frac{2.405\sqrt{\pi}}{3} h\right)}, \quad h = \frac{1}{n-1}$$

and since

$$\omega_{opt} = 1 + \lambda_{max},$$

then we can say that

$$\lambda_{max}(SBOR) \geq \lambda_{max}(SLOR) \quad \text{at} \quad \omega = \omega_{opt}.$$

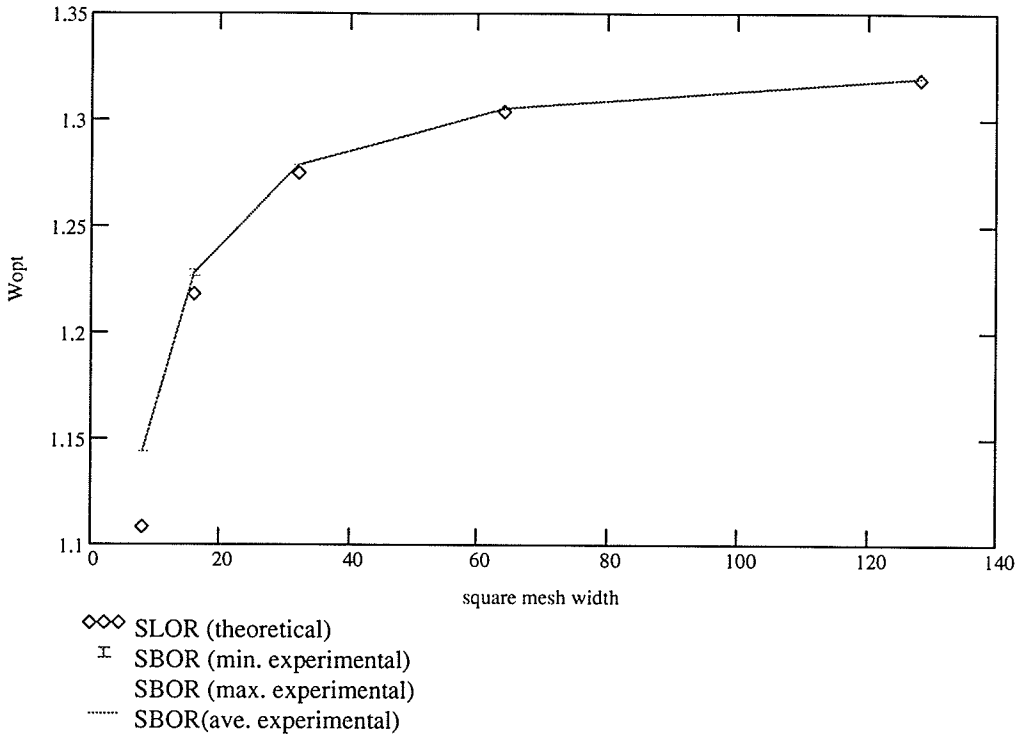


Fig. 4.16: Experimental ω_{opt} range for SBOR vrs theoretical ω_{opt} for SLOR.

In fact, λ_{max} for SBOR is just slightly larger than λ_{max} for SLOR. *It should be noted that the experimental values of ω_{opt} for SBOR are much closer to the theoretical values predicted for SLOR than are the experimental values of ω_{opt} for SLOR!* Since no published formula for ω_{opt} for SBOR is available, it was decided to slightly under-estimate it using the formula for SLOR due to the close match. Thus we can also conclude that convergence rate of h^2 for SBOR in Table 2.1 is wrong: it actually follows the $2\sqrt{2}h$ rate for SLOR. Since SBOR using 3×3 blocks was not implemented, we cannot estimate its true convergence rate.

4.4.2.3 Convergence Times

For an iteration method m, having experimentally determined the ω_{opt} and N_{min} for a 128×128 mesh and the execution time $t_{m,125}$ for a 125×125 mesh, it is possible to

determine the **convergence time** $t_{m,128}$ required to achieve convergence for a 128×128 mesh. Assuming k=1 in Eq. 4.5, we can estimate the execution time for each global sweep on the larger mesh by using

$$t_{m,128} = \left(\frac{128}{125}\right)^2 t_{m,125} \tag{4.6}$$

The convergence time $t_{m,\omega opt}$ is then computed as

$$t_{m,\omega opt} = t_{m,128} \times N\left(\omega_{opt}\right) \tag{4.7}$$

Figure 4.17 shows the computed convergence times for all the iterative methods for 64×64 and 128×128 meshes. It surprisingly indicates that *all* the accelerated block methods out-perform the accelerated point methods. This also indicates that additional computational effort required by SBOR is more than offset by its faster convergence rate compared to that of point SOR. In selecting an iterative method for solving the potential field for the dielectric breakdown model:

1. if the maximum convergence rate within a fixed number of iterations is required, Table 2.1 should be consulted for the maximum convergence rates,
2. if a variable number of iterations is allowed and the fastest convergence in the shortest time is paramount, Fig. 4.17 should be consulted for the shortest convergence times,
3. alternatively, if the fastest convergence rate within the least amount of time is required, Table 2.1 and Fig. 4.17 should be both consulted.

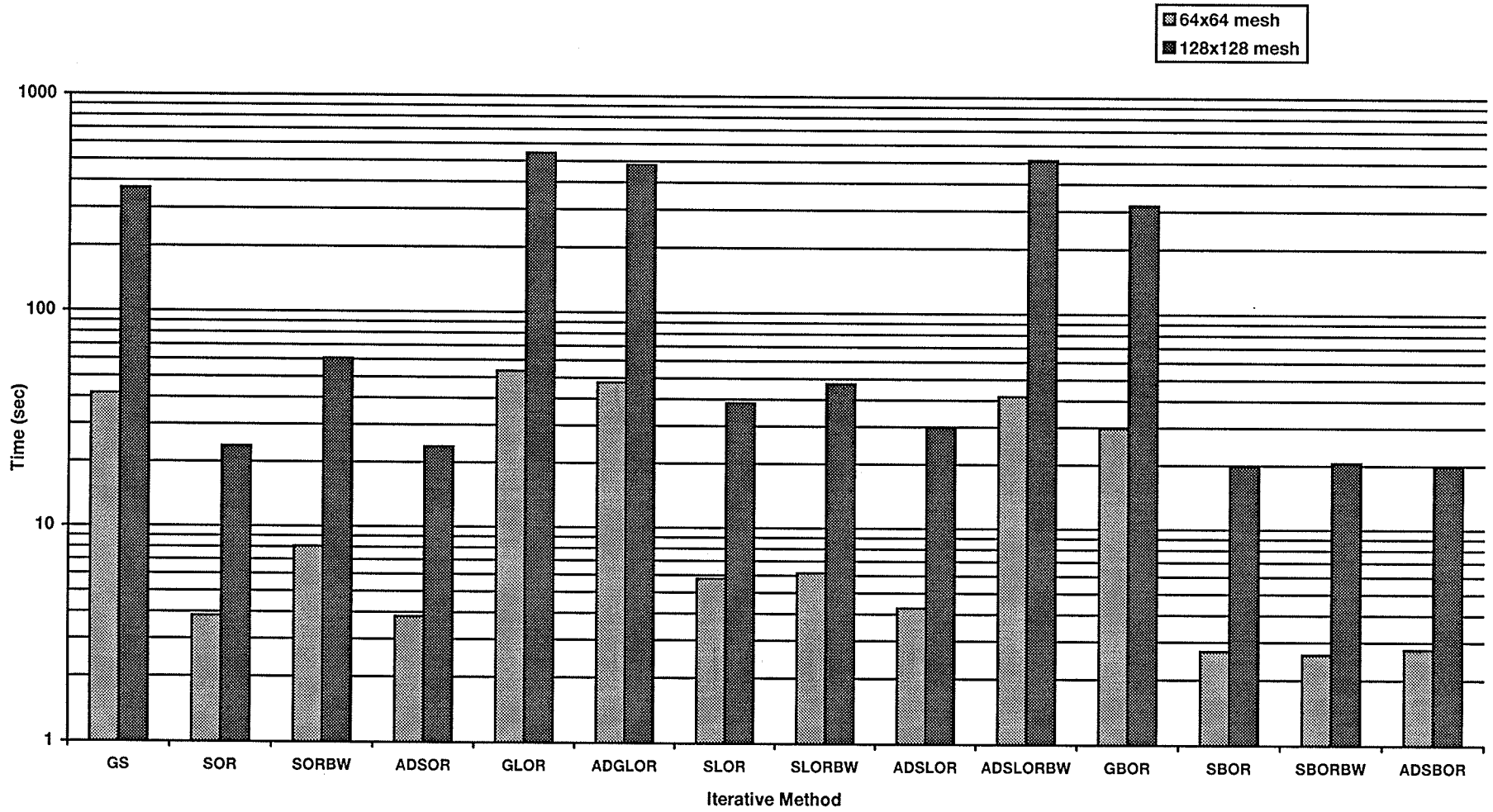


Fig. 4.17: Convergence times for iterative methods.

4.5. Experiment 4: Effect of Initialization on Experimental ω_{opt}

Experiment 4 was performed to determine if ω_{opt} for a given square mesh was independent of the initialization of the interior mesh points. The same experimental setup (Fig. 4.6) and methodology (Sec. 4.4) for experiment 3 was used for a 128×128 mesh, with the only exception being that all the interior mesh points (x,y) were initialized to a random potential $V_{x,y}$ using the *Rand2* generator such that

$$0.05 \times MaxV \leq V_{x,y} \leq 0.95 \times MaxV \tag{4.8}$$

4.5.1 Results

Figures 4.18 to 4.20 show the N versus ω curves for SOR, SLOR and SBOR. Using random initialization appears to affect the ω_{opt} of most of the accelerated iterative methods examined in this thesis. The affects can be summarized as

- *all* accelerated point methods have a minimum number of iterations N_{opt} at ω = 1.80000, which is significantly less than that for a constant initialization.
- the ω_{opt} of SLOR and ADSLOR appears to be unaffected,
- SLORBW has an ω_{opt} at 1.3000, ADSLORBW has an $\omega_{opt} \geq 1.25000$,
- *all* accelerated block methods have an N_{opt} at ω = 1.3000, an N slightly less that obtained for a constant initialization.

Why a random initialization would cause all accelerated points to have an identical optimal acceleration factor, or likewise for all accelerated block methods, or require significantly fewer iterations N for $\omega << \omega_{opt}$ is a mystery. It should be noted that these minima appear to be transients in the N vrs ω curves, hinting this is anomalous behaviour.

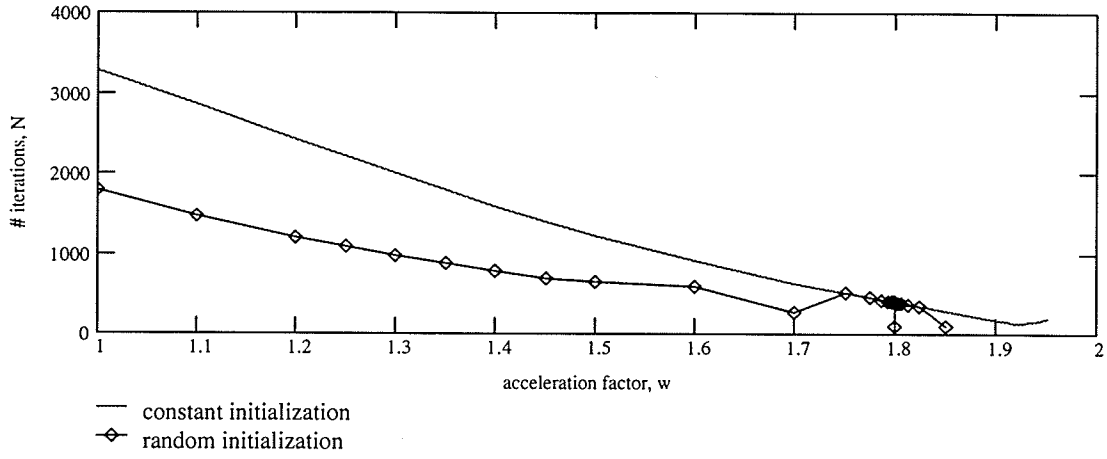


Fig. 4.18: Effect of random initialization on N vrs ω_{opt} for SOR.

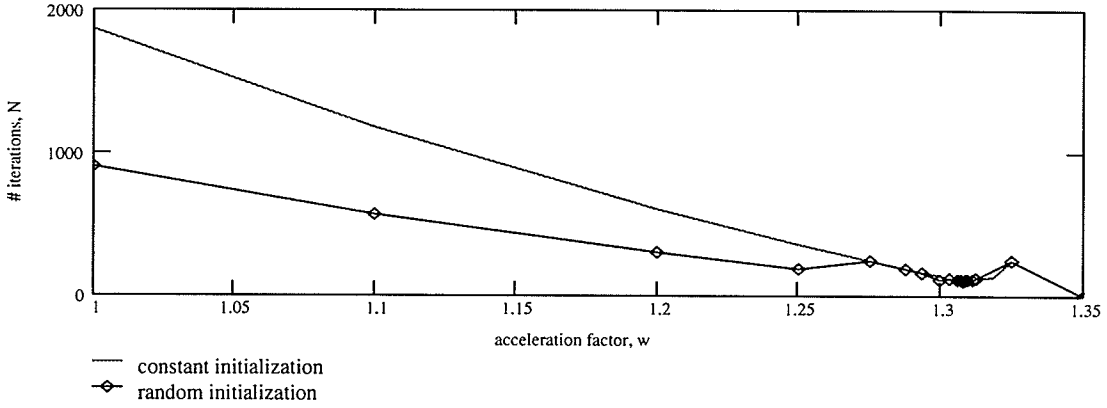


Fig. 4.19: Effect of random initialization of N vrs ω_{opt} for SLOR.

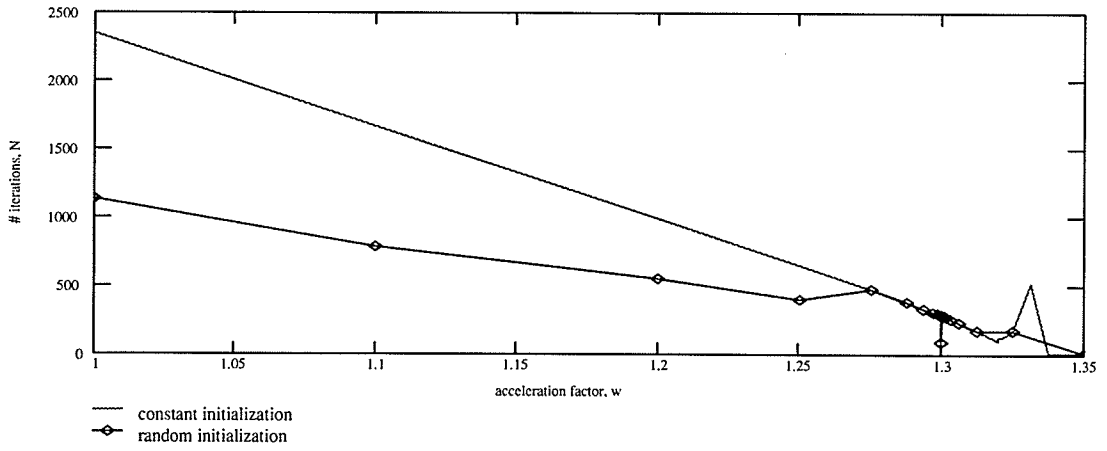


Fig. 4.20: Effect of random initialization on N vrs ω_{opt} for SBOR.

4.6. Experiment 5: Effect of Rectangularity on λ_{max} and ω_{opt}

Experiment 5 was performed to examine the behaviour of ω_{opt} for rectangular meshes as most published formulas for SLOR and ADSLOR assume a square mesh. The same experimental setup (Fig. 4.6) and methodology (Sec. 4.4) for experiment 3 was used for additional mesh sizes. Tables 4.3 and 4.4 contain the numbering of the mesh sizes. The results for square meshes from experiment 3 were used for comparison. Fig. 4.21 shows a zoomed in view of the ω_{opt} regions for the 6 test configurations for SOR.

Table 4.3. Numbering of rectangular shapes to test SOR, ADSLOR, and SBOR.

Test Shape	Width (w)	Height (h)
1	8	8
2	16	16
3	32	32
4	32	128
5	128	32
6	64	64
7	64	128
8	128	128

Table 4.4. Numbering of shapes used to test SLOR.

Test Shape	Width (w)	Height (h)
1	8	8
2	8	16
3	16	8
4	16	16
5	32	8
6	8	32
7	32	32
8	32	128
9	128	32
10	64	64
11	64	128
12	128	128

4.6.1 Impact on ω_{opt} for SOR

Figure 4.21 shows a zoomed in view of N vrs ω_{opt} for SOR. Figure 4.22 contrasts the experimental ranges of ω_{opt} with the theoretical values predicted by Eq. 4.9 for a rectangular mesh of size $w \times h$.

$$\lambda_{Jacobi} = \frac{\cos\left(\frac{\pi}{w-1}\right) + \cos\left(\frac{\pi}{h-1}\right)}{2}, \quad \omega_{opt} = \frac{2}{1+\sqrt{1-\lambda^2_{Jacobi}}} \tag{4.9}$$

Eqn. 4.9 actually over-estimates the experimental ω_{opt} by 2.77% for shapes 4 and 5, and by 1.43% for shape 6. Since over-acceleration of SOR can lead to complex eigenvalues resulting in oscillations, the estimated ω_{opt} is multiplied by 0.98 which slightly under-estimates the actual ω_{opt} for the larger mesh sizes. Figure 4.22 shows how the adjusted ω_{opt} compares with the ideal and experimental values.

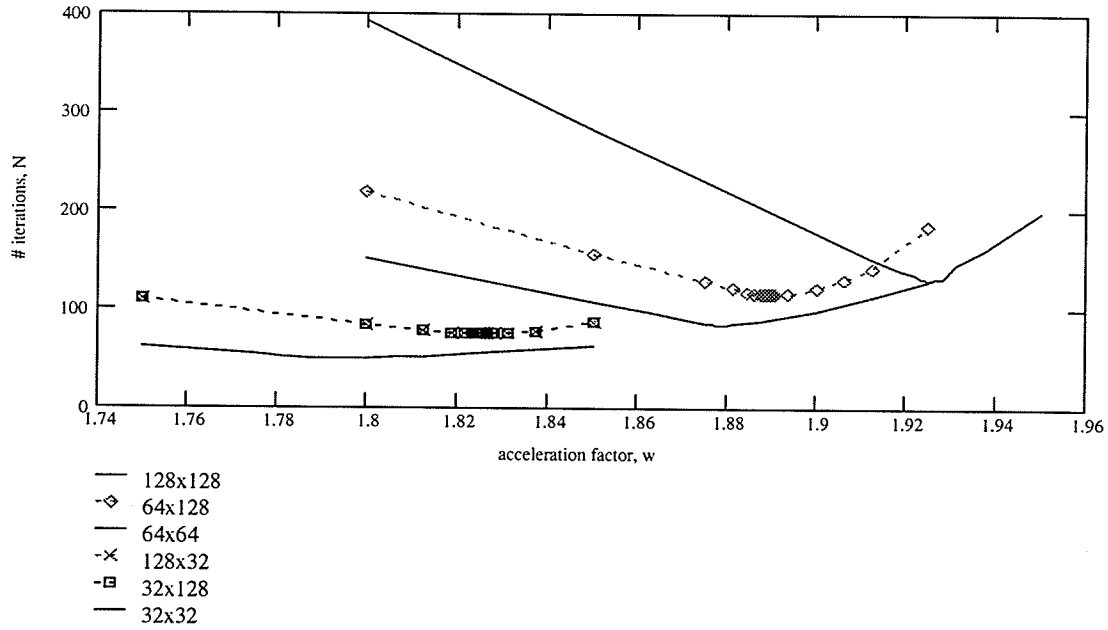


Fig. 4.21: Effect of rectangle dimensions on ω_{opt} for SOR.

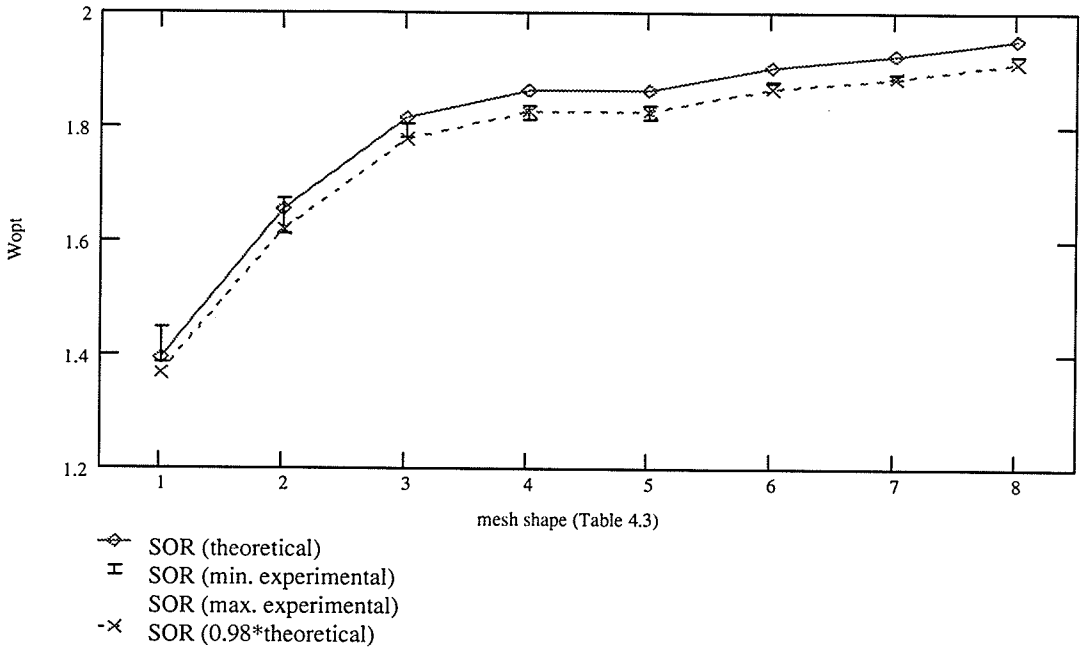


Fig. 4.22: Ideal, adjusted, and experimental ω_{opt} for SOR.

4.6.2 Impact on ω_{opt} for SLOR

Eqn. 4.10 is the formula for ω_{opt} developed by Evans [Evan62] for SLOR. The formula has a term A which is the *area* of the discrete mesh. By convention, formulas for λ_{max}, ω_{opt}, and h (the *uniform* mesh point spacing) are related to $\pi \times \pi$ or unit (1×1) mesh squares consisting of $n \times n$ points.

$$\omega_{opt} = \frac{4}{3\left(1 + \frac{2.405}{3}\sqrt{\frac{\pi}{A}}h\right)} \tag{4.10}$$

Suppose we wish to apply SLOR to rectangular regions consisting of $m \times n$ mesh points. This poses the problem that if we retain A normalized to 1.0, then an anisotropic mesh spacing is required such that

$$w = dx = \frac{1}{m-1}, \quad h = dy = \frac{1}{n-1}, \quad w \neq h \tag{4.11}$$

for which Evans has not made any provisions for in Eq. 4.10. A first attempt at modifying Eq. 4.10 involved replacing h with the geometric mean of h and w and normalizing A to the maximum of the width m or the height n of the mesh so that

$$A = \frac{m \times n}{\max(m,n)} \leq 1.0, \quad h' = \sqrt{wh} = \sqrt{\frac{1}{m-1}\frac{1}{n-1}} \tag{4.12}$$

resulting in

$$\omega_{opt} = \frac{4}{3\left(1 + \frac{2.405\sqrt{\pi}}{\sqrt[3]{\frac{mn}{\max(m,n)^2}}}\sqrt{\frac{1}{m-1}\frac{1}{n-1}}\right)} \tag{4.13}$$

As evident from Fig. 4.23, Eq. 4.13 over-estimates ω_{opt}.

A further examination of Eq. 4.10 resulted in the realization that A need not be normalized to 1, m or n. If we define a uniform mesh spacing of

$$w = h = 1, \quad \text{then} \quad A = mn \tag{4.14}$$

resulting in a better estimate of ω_{opt} described by

$$\omega_{opt} = \frac{4}{3\left(1 + \frac{4\sqrt{\pi}}{3\sqrt{mn}}\right)} \tag{4.15}$$

The *2.405* in Eq. 4.13 is replaced by *4* which tunes Eq. 4.15 to just slightly under-estimate ω_{opt} at the larger mesh sizes, resulting in a much better fit (see Fig. 4.23).

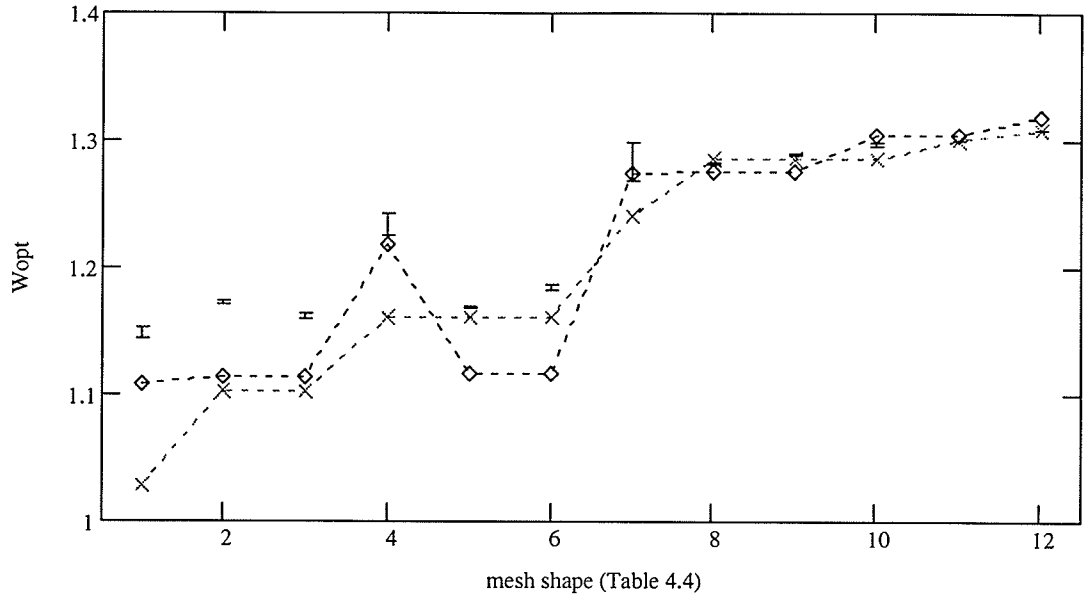


Fig. 4.23. Estimated and experimental values of ω_{opt} for SLOR.

4.6.3 Impact on ω_{opt} for ADSLOR

Eqn. 4.16 is the formula for ω_{opt} developed by Evans [Evan62] for ADSLOR which as Eq. 4.10 makes no provision for rectangular meshes.

$$\omega_{opt} = \frac{4}{3\left(1+\frac{2.405}{3\sqrt{2}}\sqrt{\frac{\pi}{A}}h\right)} \tag{4.16}$$

By the same reasoning of Sec. 4.6.2, the first estimate of ω_{opt} for ADSLOR was

$$\omega_{opt} = \frac{4}{3\left(1+\frac{2.405\sqrt{\pi}}{3\sqrt{2}}\sqrt{\frac{mn}{\max(m,n)^2}}\sqrt{\frac{1}{m-1}\frac{1}{n-1}}\right)} \tag{4.17}$$

and similarly, an improved estimate of ω_{opt} for ADSLOR which defines h=1, and which requires no "tuning" can be described by

$$\omega_{opt} = \frac{4}{3\left(1+\frac{4\sqrt{\pi}}{3\sqrt{2}\sqrt{mn}}\right)} \tag{4.18}$$

Both methods generally under-estimate ω_{opt} preventing numerical oscillations. As evident from Fig. 4.24, Eq. 4.18 provides an estimate closer to the experimental values. Due to time constraints, however, only Eq. 4.17 was implemented.

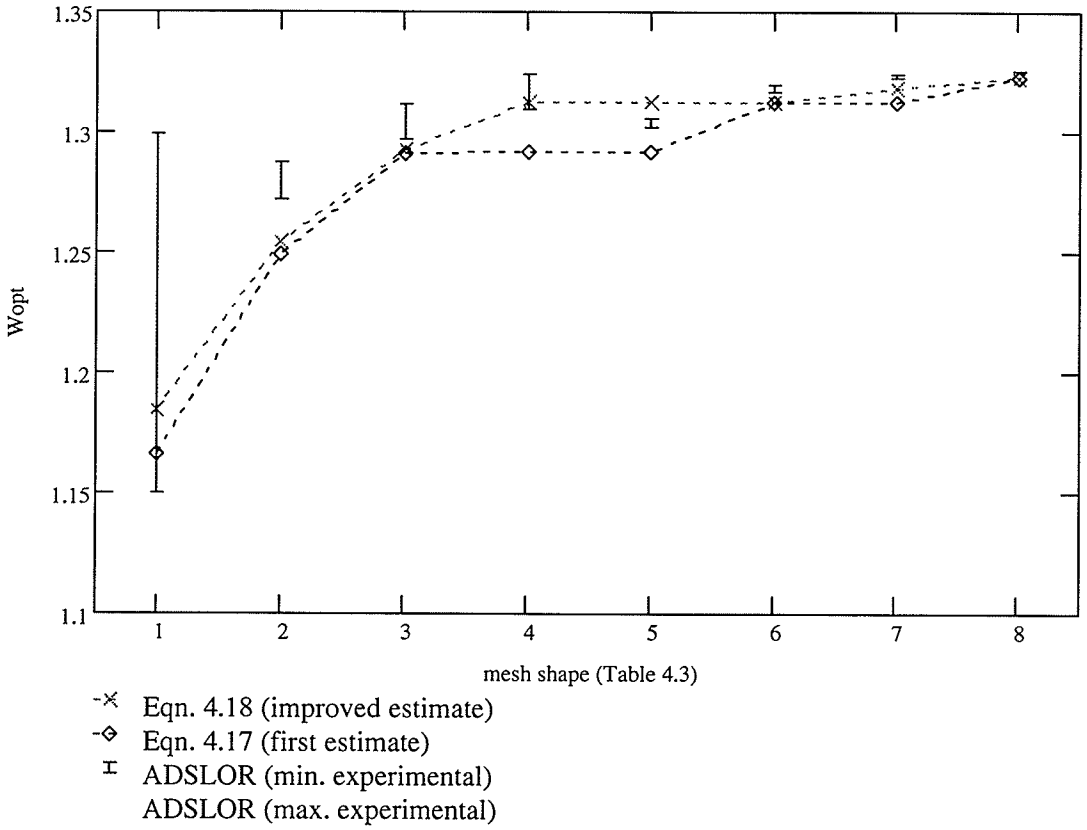


Fig. 4.24. Estimated and experimental values of ω_{opt} for ADSLOR.

4.6.4 Impact on ω_{opt} for SBOR

Since the experimental ω_{opt} of SBOR is very close to the experimental and theoretical values (from Eq. 4.10) of ω_{opt} for SLOR for *square meshes*, the same formula used to compute the acceleration factor ω_{opt} for SLOR for *rectangular mesh shapes* (Eq. 4.13) was also used for SBOR. Figure 4.25 shows that Eq. 4.13 under-estimates ω_{opt} for SBOR for the region sizes tested. A corona simulation run for a parallel plate topology employing SBOR and accelerated according to Eq. 4.13 resulted in the potential of the mesh points along and near the Neumann boundaries going to 0.0 volts. This anomalous behaviour found to be due to the incomplete, and hence incorrect, coding for stencils having Neumann boundary conditions.

Figure 4.25 also shows that the improved but unimplemented formula for SLOR (Eq. 4.15) has a poorer fit of the experimental values of ω_{opt}. If Eq. 4.13 does in fact lead to complex eigenvalues and oscillations, then Eq. 4.15 might be the formula to use.

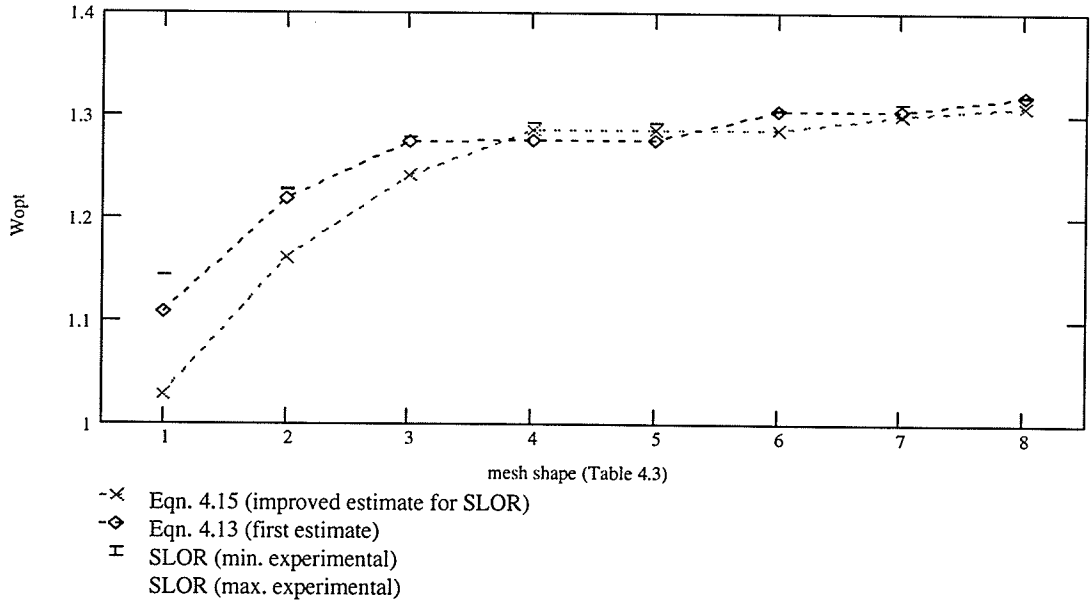


Fig. 4.25: Estimated and experimental values of ω_{opt} for SBOR.

4.6.5 Conclusions

Compared to experimental results, all the initial formulas for ω_{opt} for rectangular regions tended to over-estimate ω_{opt} which could have resulted in numerical oscillations in the mesh potentials due to complex eigenvalues. The simulations employing SLOR, ADSLOR and SBOR were performed using Eqns. 4.13, 4.17, and 4.13 respectively, and not the adjusted, improved formulas (Eqns. 4.15 and 4.18). It appears that the simulations employing SLOR and ADSLOR were unaffected by this, but the simulation employing SBOR definitely experienced some numerical problems. Slightly modifying the applicable formulas allowed us to better match the experimental results and to slightly under-estimate ω_{opt}, thereby increasing the confidence in the accuracy and behavior of the estimates of ω_{opt}.

It should be stated emphatically here that the adjusted formulas for SOR, SLOR, ADSLOR, and SBOR are valid only for singularity-free rectangular regions. *The presence of a single corona or a single electrode mesh point (i.e. a singularity) within the rectangular (or square) region of interest will affect the eigenvalues of the iterative process for that region to the degree that the adjusted equations will generate grossly wrong acceleration factors.*

4.7. Experiment 6: Evaluation of Norm-based Estimates of λ_{max}

Experiment 6 was performed to evaluate the performance of the classical approach to estimating the spectral radius λ_{max} of the iteration process from the 1^{st}, 2^{nd} or infinite residual norms (Eqns. 2.76) such that

$$\lim_{k\to\infty}\frac{\left\|\mathrm{r}^{k}\right\|_{p}}{\left\|\mathrm{r}^{k-1}\right\|_{p}}=\lambda_{\max},\quad \text{for any norm } p=1,2,\infty \tag{4.19}$$

To distinguish between the 3 estimates of λ_{max}, the following notation will be adhered to:

- λ_1 ($L1$, or L_1) is the 1st residual norm-based estimate of λ_{max},
- λ_2 ($L2$, or L_2) is the 2nd residual norm-based estimate of λ_{max}, and
- λ_∞ ($L\infty$, or L_∞) is the infinite residual norm-based estimate of λ_{max}.

4.7.1 Setup

The experimental setup consisted of a parallel plate electrode topology modeled on a 256×256 mesh (Fig. 4.26). A corona discharge was allowed to grow upwards from the bottom plate until it reached the top plate (Sec. 2.3.1). After each growth stage, 50 local Gauss-Seidel passes were performed within a 51×51 local region centered at the most recent growth site. This was followed by 100 global point Gauss-Seidel passes on the entire mesh. The 3 residual norms were computed after each global pass and saved to a (huge) file. The standard mesh scanning order (Sec. 4.2.2) was used for all iteration passes. The tag array was scanned in a reserve row-order fashion (Sec. 4.2.3). Figure 4.27 shows the 3 residual norms for the 100 global sweeps associated with the 1st, 2nd and 3rd growth stages.

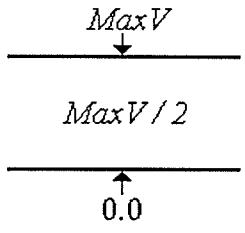


Fig. 4.26: Mesh initialization for experiment 6.

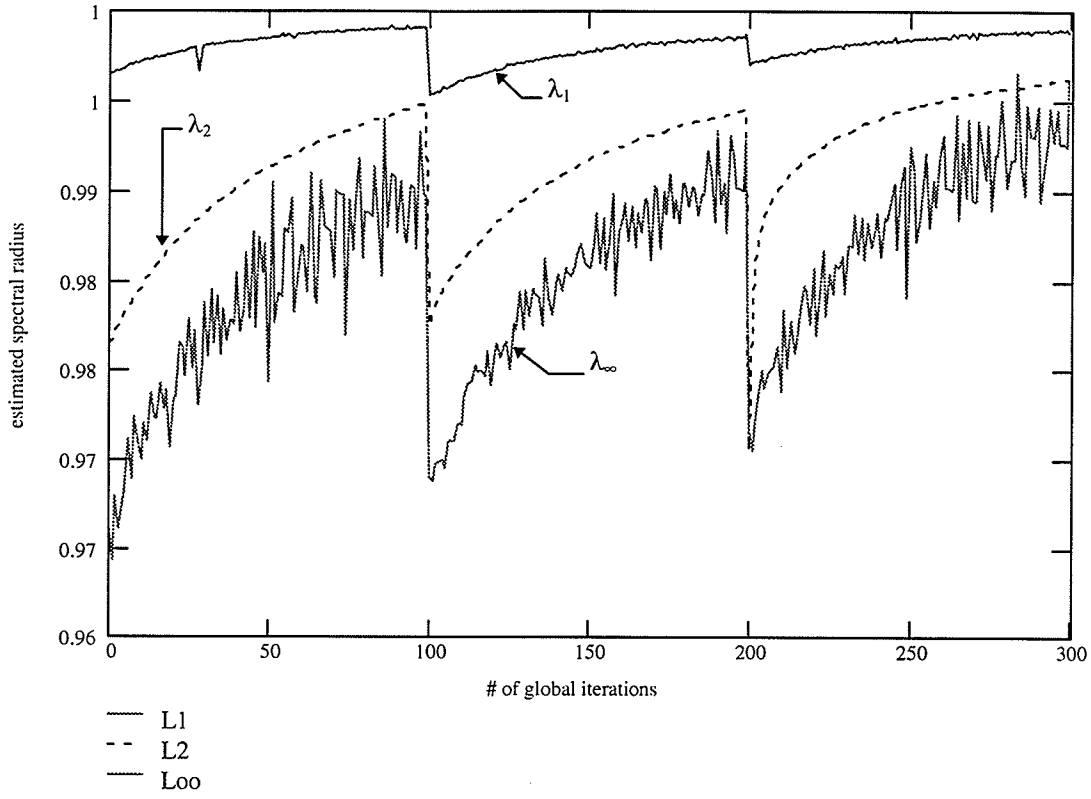


Fig. 4.27: λ_1, λ_2, and λ_∞ estimates of λ_{max} for first 3 growth stages.

4.7.2 Discussion

Assuming a worst case situation of pure vertical corona growth, after the 3rd growth stage, there would remain a singularity-free, interior rectangular region of at least 250×250 points which has a minimum spectral radius $\lambda_{max} \geq 0.999842$ for Gauss-Seidel. From Fig. 4.27, it can be seen that λ_1 is the closest estimate to this value.

Due to the many wavefronts of numerical correction simultaneously "washing" over the mesh and reflecting back from the Dirichlet and Neumann boundaries, the infinite norm is not necessarily monotonically decreasing. This results in the "rough", non-monotonically increasing curve for λ_∞, which turns out to be the worst possible

estimate of λ_{max}. Since this is also characteristic of *accelerated methods*, any estimate of the optimal acceleration factor ω_{opt} based on the λ_∞ estimate will therefore be the worst possible estimate of ω_{opt}, leading to the conclusion that the infinite norm-based estimate of λ_{max} should *never* be used to compute the acceleration factor.

The λ_2 estimate lies between the λ_1 and λ_∞ estimates, but is still grossly wrong compared to λ_1. It is roughly estimated that

- at least 300 global iterations would be required for the λ_2 estimate to approach the same order of accuracy attained by the λ_1 after only 100 global iteration passes, and
- at least 1000 global iterations would be required for the λ_1 estimate to approach 0.999842 within 1%.

Due to the close spacing of the eigenvalues associated with all the iteration methods evaluated, all the eigenvalues initially contribute to the reduction of the error in the mesh potentials, so that the error reduction is greater that what would be due to the maximum eigenvalue (λ_{max}) alone. This is advantageous from a convergence viewpoint, however it reduces the estimate of λ_{max}, negatively impacting the estimate of ω_{opt}. As the number of global iterations increases, the effects of the sub-dominant eigenvalues die out, and the error reduction rate approaches λ_{max} from below.

The characteristics observed in the first 3 growth stages are also consistent throughout the entire growth process for this experiment, which does not exceed 6000 growth stages (each growth stage adds one mesh point to the discharge). Displaying the graphed results for 600,000 global iterations is impractical due to clarity and reproduction concerns.

4.8. Improving the Estimation of λ_{max} and ω_{opt}

Given the poor norm-based estimation (Sec. 4.7) of λ_{max} and hence also of ω_{opt}, it was decided to re-examine the estimates of λ_{max} based on the adjusted classical formulas (Eqns. 4.9, 4.15, 4.18) and based on the classical ratios of successive norms (Eq. 4.20). This re-evaluation lead to the development, implementation and evaluation of an improved method of estimating λ_{max} or ω_{opt} at each growth stage apriori to any global iterations.

4.8.1 Classical Formula-based Estimates

4.8.1.1 Advantages

The classical formulas for λ_{max} and ω_{opt} for point, line and block iterative methods operating on singularity-free square meshes have the advantage that they are easily calculable apriori of any global iteration sweeps so that ω_{opt} is immediately known. Furthermore, these ω_{opt} estimates remain constant, unlike the norm-based estimates which ramp up towards the unknown exact value of ω_{opt}.

4.8.1.2 Disadvantages

When it comes to estimating ω_{opt} in our situation in which the interior mesh points are infected with singularities due to the growing corona discharge, we find there are many disadvantages and shortcomings to employing the classical formulas. These are listed below.

- Classical formulas for λ_{max} and ω_{opt} have been derived only for very simple mesh shapes. Formulas for SOR address rectangular shapes, and formulas for SLOR, ADSLOR and SBOR address only square mesh shapes, but they have been modified

to also include rectangular shapes (Sec. 4.6) as part of this thesis. Estimating λ_{max} for circular shapes analytically is difficult: it can be estimated from an equivalent rectangle of equal area [Rand67], by a membrane analogy [LlMc68], and by the smallest enclosing square [Lewi71], to name a few methods. Estimates of λ_{max} for complex shapes are generally unknown or intractable.

- Estimates of λ_{max} and ω_{opt} from classical formulas are seriously affected by the introduction of *even a single* singularity in the mesh interior, and the corona discharges simulated on 512×512 meshes can inject up to 25,000 singularities. The numerical methods developed by Motz [Motz47] and others [BeCr73] to handle enclosed singularities are not practical in our case considering the large number and random locations of the corona points.
- Classical formulas for λ_{max} and ω_{opt} assume $N \times N$ or $N \times M$ mesh sizes consisting of $(N\text{-}2)\times(N\text{-}2)$ or $(N\text{-}2)\times(M\text{-}2)$ interior mesh points surrounded on all 4 sides by Dirichlet boundaries. The introduction of Neumann boundaries along the vertical sides in the parallel plate topology increases the spectral radius and optimal acceleration factor for all point, line and block iterative methods. Figure 4.28 shows how the introduction of Neumann boundaries along the vertical mesh sides increases the spectral radius for the point Jacobi and SOR methods. As the mesh size increases, λ_{max} for Neumann boundaries approaches the λ_{max} for pure Dirichlet boundaries from above. Thus, any formula which *exactly predicts* λ_{max} for pure Dirichlet boundaries will *slightly under-estimate* λ_{max} for the parallel plate topology.

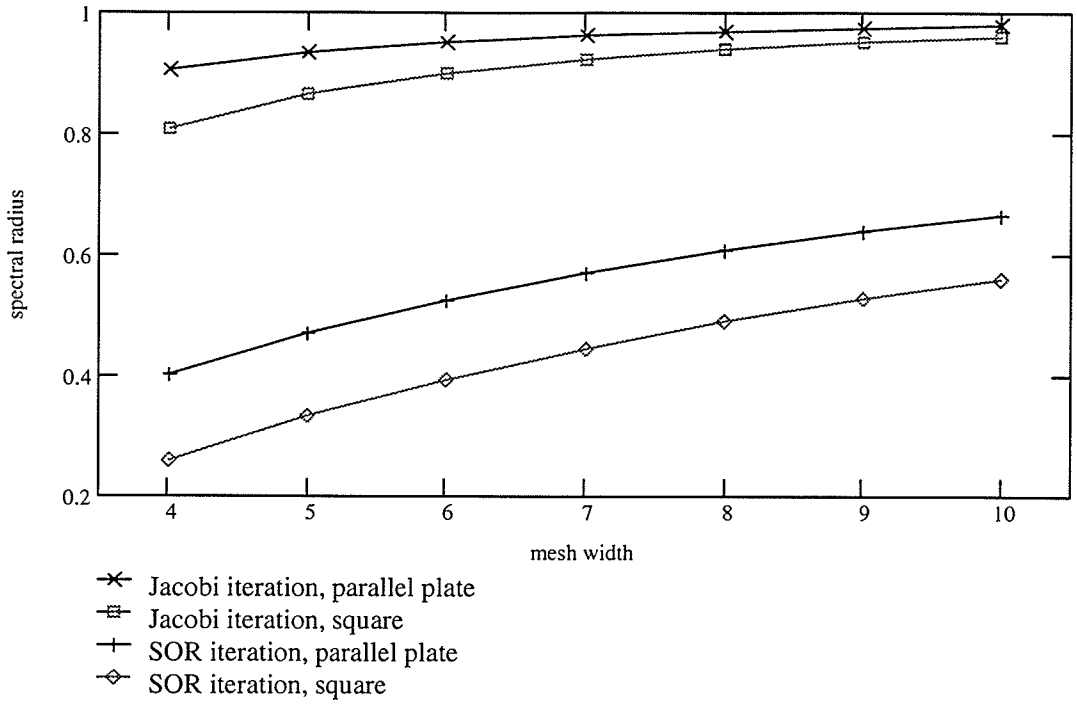


Fig. 4.28: Effect of Neumann boundaries on λ_{max} for various square mesh sizes.

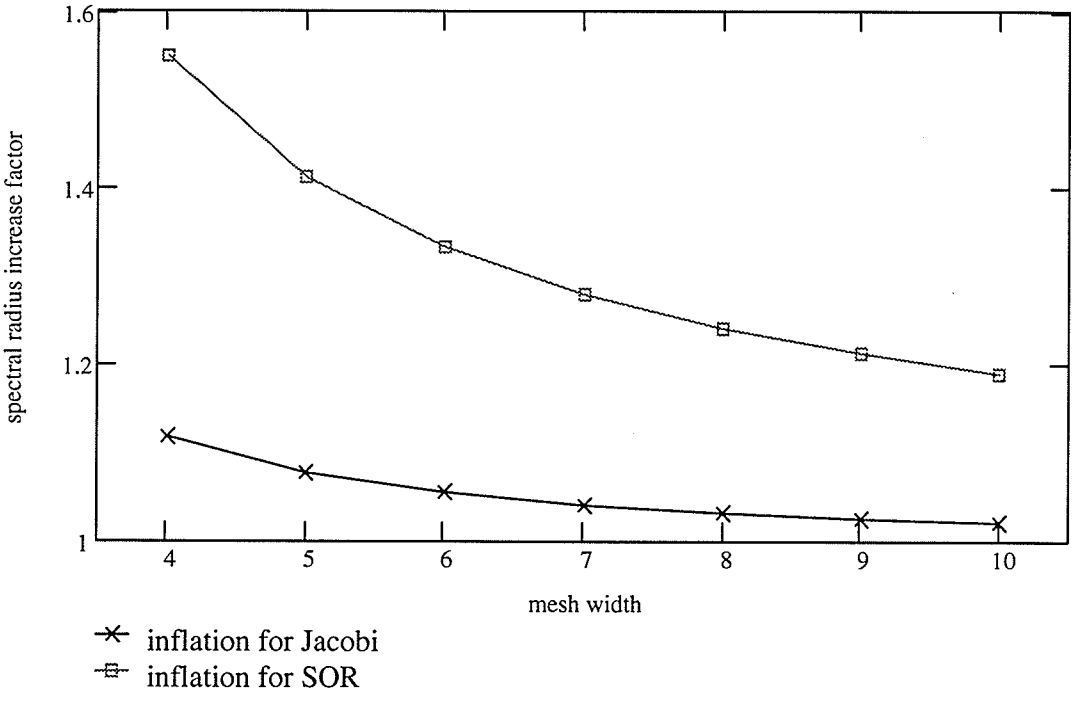


Fig. 4.29: Inflation of spectral radius due to Neumann boundaries.

4.8.2 Classical Norm-based Estimates

4.8.2.1 Advantages

The main advantage offered by the classical norm-based estimates of λ_{max} and ω_{opt} is that they take into account the global configuration and characteristics of the mesh and the global impact of the iterative process. Thus, they are well suited, and in many cases the *only* method available (until now), to estimate λ_{max} when:

- the mesh is a complex shape,
- there are any number of singularities in the mesh interior, or
- the mesh has any number of Neumann boundaries along mesh edges.

4.8.2.2 Disadvantages

As seen in Fig. 4.27, norm-based estimates of λ_{max} require a considerable number of global iterations to converge to λ_{max}. If t is the number of global iteration passes, the estimates are *exact* only at $t \rightarrow \infty$, and a large t is required for even *reasonably close* estimates of λ_{max}. Following previously developed dielectric breakdown models [Aren94, Stac95], only 8 global iteration passes are performed after each growth stage, thus any norm-based estimate of λ_{max} will grossly under-estimate λ_{max} resulting in an under-estimated ω_{opt} and a reduced convergence rate. The end result is that the global potential field will be much more accurate than if point Gauss-Seidel were used, **but** not as accurate as it could be had the *optimal* acceleration factor ω_{opt} been used.

4.8.3 The Improved Estimation Method

To address the concerns raised about the suitability of classical methods of estimating λ_{max} and ω_{opt} for iterative methods under the special conditions posed by the

DBM, an improved estimation was developed. The required theory is presented first.

4.8.3.1 Theory

Prior to presenting the theorems under-pinning the improved estimation methods, several observations need to be stated.

Observation 1:

The spectral radius for any singularity-free rectangular region A is immediately known and calculable using the classical or improved formulas.

Observation 2:

Adding a *complete row* or a *complete column* to a singularity-free rectangular region A can only increase spectral radius. This is self-evident from the classical and improved formulas.

Observation 3:

Adding *single points* or *clusters of connected points* to the perimeter of an existing singularity-free rectangular region A will always increase the spectral radius of the enlarged region A. This characteristic is evident from experimentation.

Based on the above observations, the following two theorems are proposed.

Theorem 4.1:

Given any irregular-shaped mesh M surrounded by any combination of Dirichlet and Neumann boundaries, if any points on the periphery of M are removed so that what is left is a singularity-free rectangular region A, then the exact spectral radius of that rectangular region A will be less than the exact spectral radius of the original mesh M.

Figure 4.30 illustrates Theorem 4.1. The relationship between the spectral radius $\lambda_{max(M)}$ of the irregular mesh shape M and the spectral radius $\lambda_{max(A)}$ an enclosed, singularity-free rectangular region A is such that $\lambda_{max(M)} > \lambda_{max(A)}$.

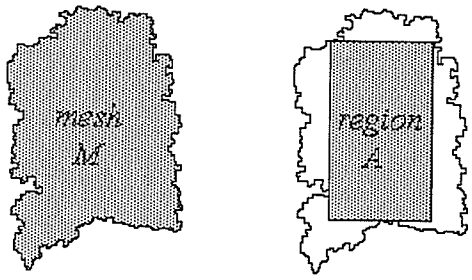


Fig. 4.30: Illustration of Theorem 4.1.

Theorem 4.2:

Given any mesh M which has a rectangular-shaped interior which

- is surrounded by any combination of Dirichlet and Neumann boundaries, and
- is infected with singularities (in the interior of the mesh),

then a good estimate λ_A of the spectral radius $\lambda_{max(M)}$ of the mesh M can be obtained by:

1. segmenting the interior of the mesh M in to n, possibly overlapping, singularity-free rectangular regions labeled as R_1, R_2,...R_n, and
2. computing according to the selected iteration method, the spectral radius of each rectangular region such that

$$\lambda_k = \lambda_{\max}(R_k), \quad k = 1,2,\ldots n \tag{4.21}$$

3. and extracting the largest computed spectral radius λ_A such that

$$\lambda_A = \max(\lambda_k), \quad k = 1,2,\ldots,n \tag{4.22}$$

Convention:

As a matter of convention, the region which satisfies 4.22 is called *region A*.

Theorem 4.3:

Within the constraints of the employed dielectric breakdown model, the largest value λ_A of the computed spectral radii of the n singularity-free rectangular regions will always be closer to the exact value λ_{max} than any classical norm-based estimates.

4.8.3.2 Implementation for Global Update Regions

The improved estimation method is implemented in a straightforward manner. Initially, prior to any corona growth, the mesh interior is singularity-free. The mesh is segmented into multiple, possibly over-lapping, contracting, singularity-free, rectangular regions (SFRRs): 4 which are anchored to the corners of the mesh (Fig. 4.31, and 4 which are anchored to the sides of the mesh (Fig. 4.31 b & c, note: the region anchored to the bottom side has vanished). Initially each region covers the entire mesh. Figure 4.31 shows the contracting SFRRs for a parallel topology and the contraction directions (indicated by arrows).

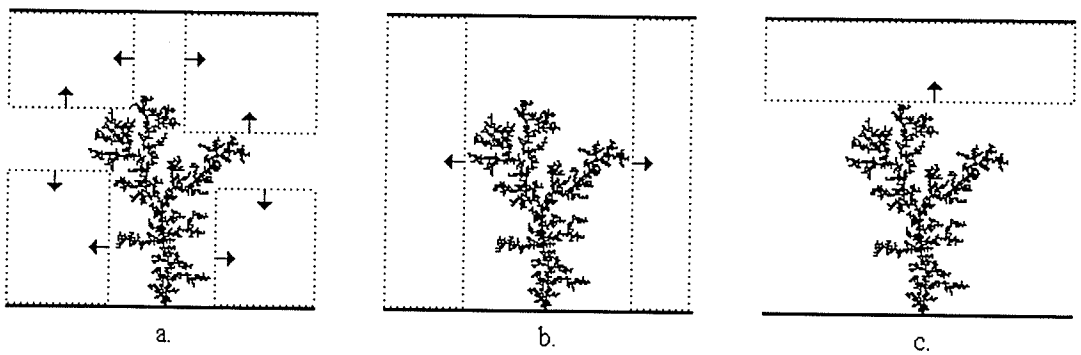


Fig. 4.31: Typical singularity-free rectangular regions used by global updates.

The use of multiple SFRRs, each contracting differently during the growth

process, increases the likelihood of obtaining the best possible maximum value for λ_A. Using values for point SOR, Table 4.5 shows that the spectral radius of a rectangular region does *not* scale with its area.

Table 4.5: SOR spectral radii of sample rectangular regions.

mesh dimensions (mesh points)			
width	height	area	λ_{max}
510	10	5100	0.666263988
15	15	225	0.673513678

After each corona growth event, each SFRR is analyzed and possibly contracted to exclude the new corona point. During the entire growth process, the area of some SFRRs may collapse totally. In effect, the SFRRs behave similar to opening (contracting) camera shutters so as to continually bound the growing and expanding discharge.

Since the SFRRs will generally be rectangular and not square, the adjusted formulas (Eqs. 4.15, 4.18) should be used so as not to over-estimate ω_{opt}. Evans' formulas for SLOR and ADSLOR compute only ω_{opt}, however, we can derive λ_{max} if required using

$$\lambda_{max} = \omega_{opt} - 1 \tag{4.23}$$

Following the computation of ω_{opt}, the 8 global iteration passes can be performed.

4.8.3.3 Implementation for Local Update Regions

After each stage of growth, the mesh potentials are perturbed: drastically in the immediate locality of the most recent growth site $P(x,y)$, and to a lesser degree globally.

To increase the rate of convergence, 50 iterations are performed within a 51×51 local update region (LUR) centered on P, prior to the 8 global iterations. Since the position and characteristics of the LUR change after each new growth stage, the singular-free rectangular regions must be determined anew, so a simpler segmentation scheme than Fig. 4.31 is used. Four SFRRs, each anchored at a different corner of the LUR (Fig. 4.32), are employed to estimate its spectral radius.

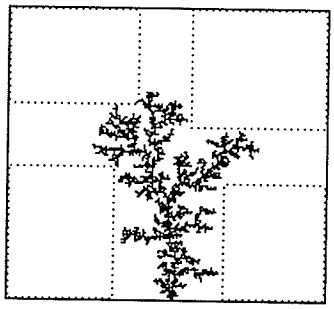


Fig. 4.32: Typical singularity-free rectangular regions used by local updates.

4.8.3.4 Advantages

The new estimation method offers the following attractive advantages over traditional norm-based estimates, in that the estimates of λ_{max} and ω_{opt}

- can be computed apriori to any global iterations, eliminating the expensive mesh rescanning (Fig. 4.33) to compute the norms after each global iteration pass.
- are constant for the 8 global iterations which follow each corona growth stage. There is no ramping of estimates similar to norm-based estimates.
- are more accurate (than norm-based estimates).

The algorithmic differences between the classical and improved method is shown in Fig. 4.33. Like norm-based methods, the new method can be utilized by all the iterative methods evaluated.

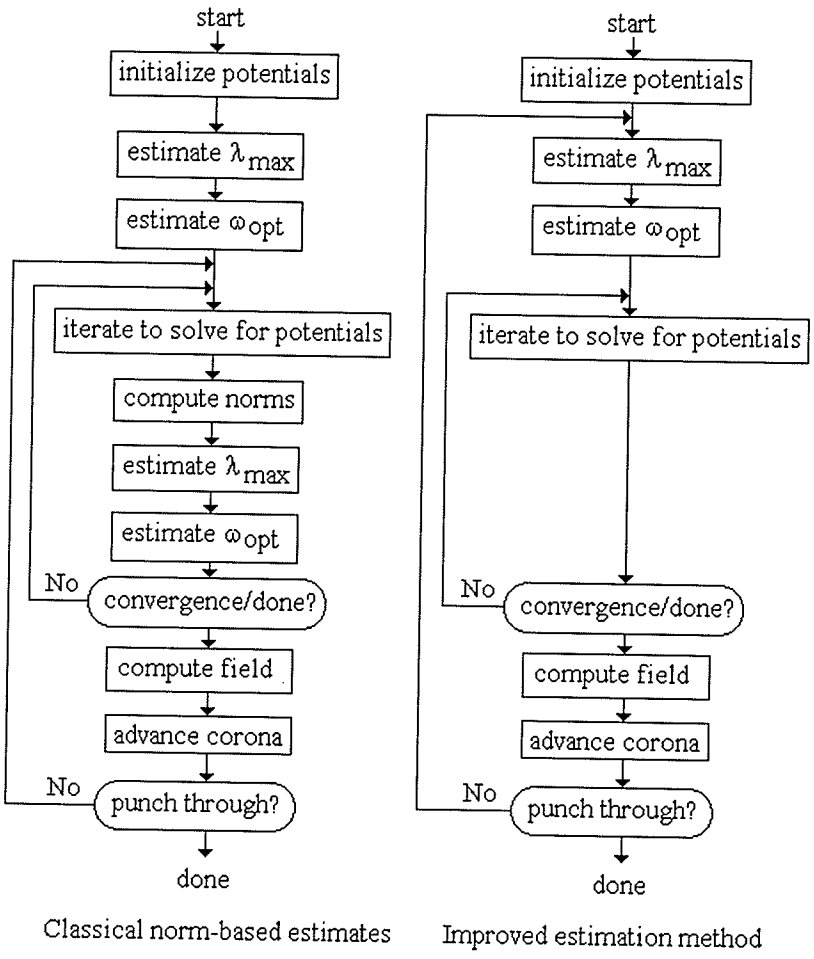


Fig. 4.33: Algorithms for classical and improved estimation methods.

4.8.3.5 Caveats

No claims are made with regards to the new estimation method:

- concerning its performance for topologies which have circular or annular-shaped interior mesh regions.

- for iterative methods other than SOR, SLOR, ADSLOR, or SBOR as these are the only methods which have been evaluated with the new estimation method.
- concerning its suitability to other field problems.
- for suitability to 3-D problems as they were beyond the scope of this thesis.

4.8.4 Experiment 7: Comparison of Estimation Methods

Experiment 7 was performed to assess the performance and impact of the improved estimation method on the iteration process and to compare them with the classical norm-based methods.

4.8.4.1 Experimental Setup

To perform the experiment, the same 510×510 mesh size and experimental configuration as experiment 6 was employed. The parameter variations for the 4 simulations appear in Table 4.6.

Table 4.6: Simulations to test impact of estimation methods.

run name	iterative technique		estimation method
[filename]	local updates	global updates	for λ_{max}, ω_{opt}
app220	Gauss-Seidel	GS	n/a
app231	Gauss-Seidel	SOR	norm-based
app230	Gauss-Seidel	SOR	improved
app330	SOR	SOR	improved

The first run (app220) was done for reference purposes. A run (app231) accelerated using traditional norm-based estimates of λ_{max} and ω_{opt}, and a run (app230) accelerated using the improved estimation method were performed to compare the

performance of the methods. The first 3 runs used Gauss-Seidel to iterate the local update regions. A fourth run (app330) was performed to determine the impact of accelerated local updates on the error reduction.

The selected measures of performance are:

- the *maximum* residual error, equal to the infinite residual norm, and
- the *average* residual error e_{ave}^{k} after the k^{th} growth stage derived from the 1st residual norm as

$$e_{ave}^{k} = \frac{1^{st}\text{ residual norm}}{\#\text{non}-\text{singular interior mesh points}} = \frac{\left\|\mathrm{r}^{k}\right\|_{1}}{(n-2)(m-2)-k} \tag{4.24}$$

where **r** is the residual vector (Eq. 2.68), the mesh size is $n \times m$, and k is the growth stage, which is also the number of corona points, or singularities in the mesh. The 1st and infinite residual norms are computed and saved after each global iteration pass.

4.8.4.2 Results

The following several pages show the performance improvements obtained when moving from non-accelerated local and global Gauss-Seidel updates to norm-based accelerated global SOR updates, to improved global SOR acceleration, to improved acceleration of SOR for local *and* global update regions.

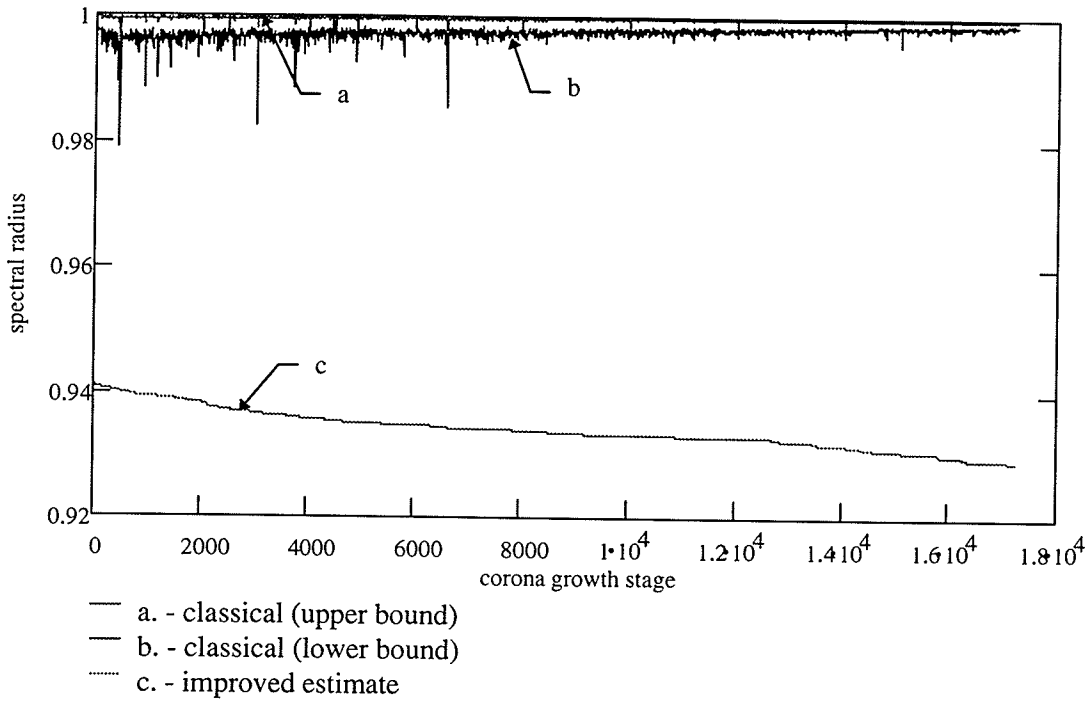


Fig. 4.34: Comparison of estimates of λ_{max} for app230.

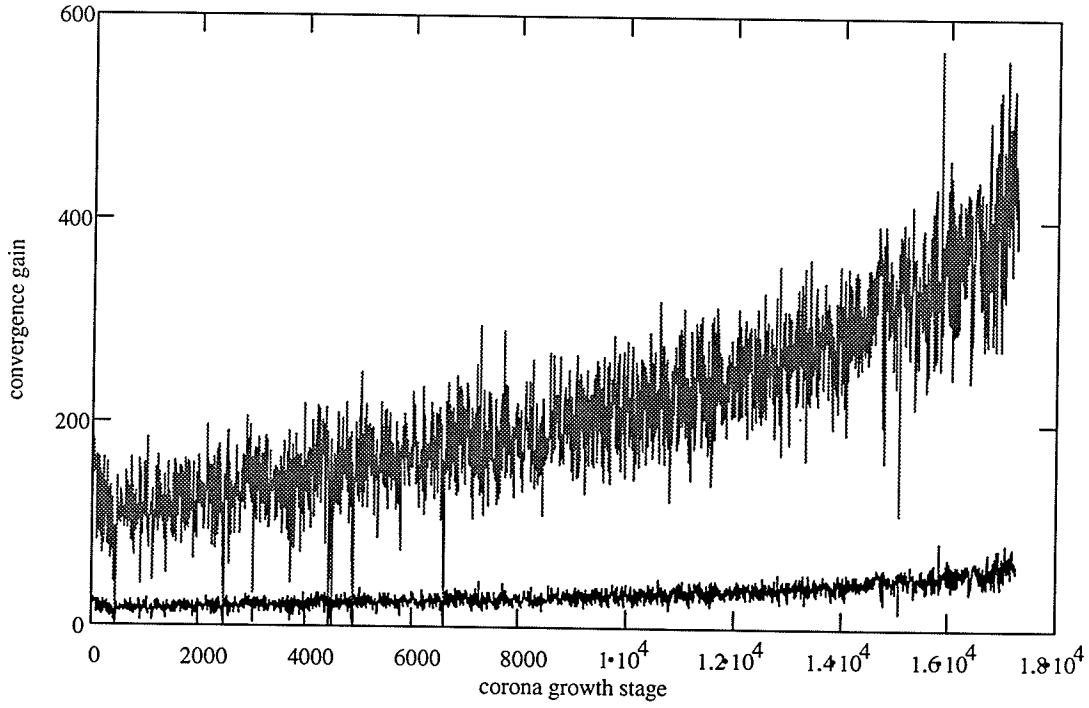


Fig. 4.35: Bounds of convergence gain due to improved estimation method.

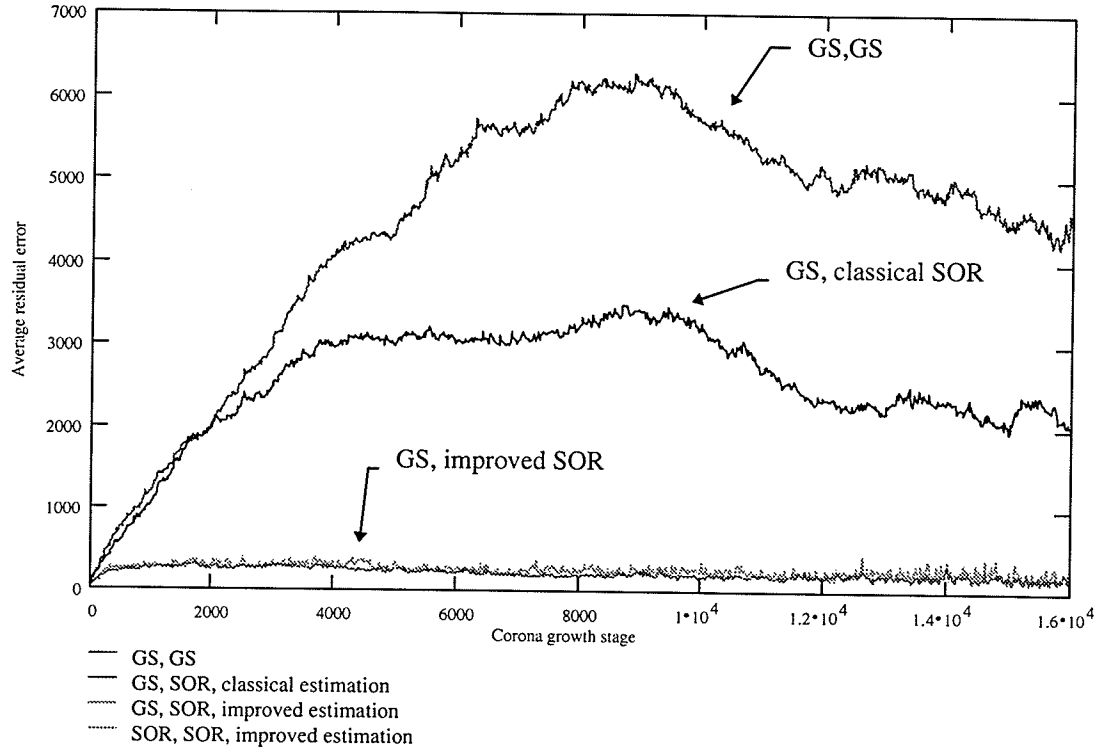


Fig. 4.36: Effect of methods on the average residual error.

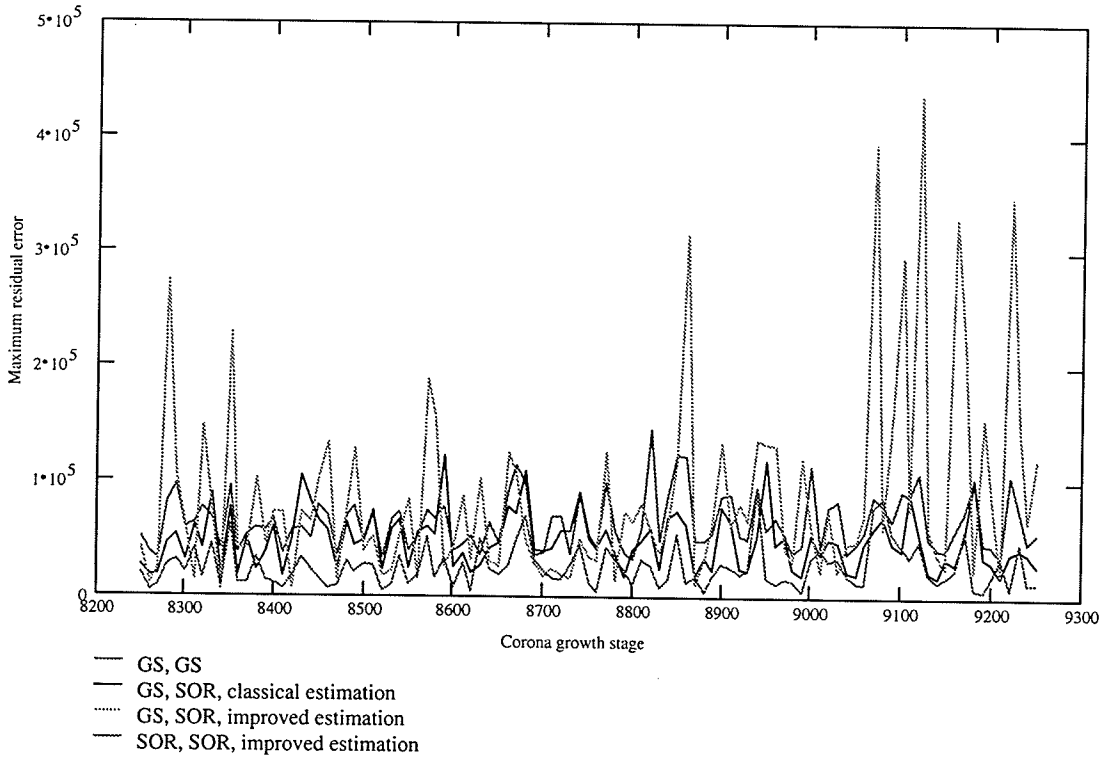


Fig. 4.37: Effect of methods on the maximum residual error.

4.8.4.3 Discussion

The data for Figs. 4.34 and 4.35 are computed from data from the **app230** simulation. Except for the first computation epoch, the *exact* spectral radius μ is unknown. During each epoch, the estimate of the Jacobi spectral radius λ_{est} is computed from the first residual norms using Eq. 2.76. The optimal acceleration factor is then estimated using Eq. 2.52. Eq. 4.25 can be used to determine the exact operating point on a λ_{max} vrs. ω curve (Fig. 4.8) if the exact Jacobi spectral radius μ and λ_{est} or the estimated acceleration factor ω are available.

$$\lambda_{\max(SOR)} = \left[\frac{\omega\mu + \sqrt{\omega^2\mu^2 - 4(\omega - 1)}}{2}\right]^2 = \left[\frac{\mu + \sqrt{\mu^2 - \lambda_{est}^2}}{1 + \sqrt{1 - \lambda_{est}^2}}\right]^2 \qquad (4.25)$$

The exact spectral radius is not known, but it will lie between the spectral radii μ_{max} for a 508×508 singularity-free mesh and μ_{min} corresponding to the 144×508 singularity-free block to the right of the final fractal. These two extrema of μ are used to compute the bounds of the SOR spectral radius shown in Fig. 4.34. Notice that the improved estimation method yields a much lower estimate. The convergence gain bounds is then the ratio of the convergence rates computed using Eq. 2.43 and shown in Fig. 4.35. The convergence rate of the new improved method is between 25 and 400 times better than that offered by the classical method of estimating the spectral radius.

The preceding graphs show a continual improvement in the reduction of the average residual errors as better iteration and estimation methods are used. Table 4.7 shows the reduction ranges of the errors relative to the errors determined for the non-accelerated Gauss-Seidel local and global iteration method.

As for the maximum residual errors, the SOR run accelerated by norm-based estimates is very similar to the non-accelerated Gauss-Seidel simulation. The best performance occurred when improved acceleration was used for both local and global iterations. The worst performance award goes to the run having local Gauss-Seidel iterations and global SOR iterations accelerated by the improved estimation method, which was characterized by transient values at least 4 times greater than values for the other methods. Figure 4.35 shows that the improved estimate method generally has higher estimates of the spectral radius which are closer to the unknown λ_{max}, resulting in an acceleration factor which is much closer to ω_{opt}, leading to better error reduction.

Table 4.7: Error reduction performance improvement due to methodology.

iterative technique		estimation of	relative error	
local updates	global updates	λ_{max}, ω_{opt}	maximum error	average error
Gauss-Seidel	Gauss-Seidel	n/a	1.0	1.0000
Gauss-Seidel	SOR	norm-based	~1.0	0.5462
Gauss-Seidel	SOR	improved	1.0 to 5.0	0.0433
SOR	SOR	improved	0.2 to 0.8	0.0305

The question that needs to be addressed is this: does the poor maximum error performance necessitate that the improved estimation method be discarded? The answer is a resounding no, since the *average error*, which is a global measure of the impact of the iterative process is reduced to 1/17 of the error for the classical norm-based acceleration. For the Gauss-Seidel/improved-SOR simulation, global performance improvement exacts a local penalty. The best reduction of the average *and* maximum residual errors occurs when improved acceleration is used for *all* iterations.

The 4 simulations performed did not result in identical discharges. This is due to the differences in the methods employed, resulting in slightly different computed potential fields, in different distributions of growth probabilities, and finally, in non-identical discharges having different numbers of corona points. This should be taken into account when comparing the side-by-side curves of error reduction data from simulation runs having differing number of points.

4.9. Multifractal Measurements of Laplacian Fractals

Unless noted otherwise, the test configuration employed to generate the following fractals consisted of:

- a 510×510 discrete mesh used to model a parallel plate topology (Fig. 4.26),
- normal mesh and tag array scanning (Secs. 4.2.2, 4.2.3)
- Gauss-Seidel used for local (50 passes on a 51×51 region) and global updates, and
- the rand2 (81) generator (App. B) used to select the growth site.

Since more information can be obtained from the Mandelbrot dimension than just from the Rényi dimensions themselves, the Rényi dimension curves are not presented here. The Mandelbrot dimensions were computed in order to *fingerprint* the Laplacian discharges. This allows us to compare the complexity and structure of the different, yet similar, discharges. Without the multifractal fingerprints or *signatures*, we would just have a library of pretty pictures. Since all the corona points are connected, unlike the isolated dust characteristic of strange attractors, there is a high degree of correlation between the corona points, so the Mandelbrot dimensions are computed using generalized correlation. A detailed analysis of the Mandelbrot dimension curves is beyond the scope

of this thesis and is left for further study. Pictures of the final potential field and the Laplacian fractals can be found in Appendix E.

4.9.1 Experiment 1: Effect of Model Parameter Variations

Figure 4.38 shows the Mandelbrot dimension f_α versus α_q curves (labelled as fa and aq) for the simulations performed to determine the effect of parameter variations (Sec. 4.2, Table 4.1). Except for the curve for **trinr4**, all the Mandelbrot dimension curves are tightly grouped together. If we compare Fig. 4.38 with Fig. 4.40, then even the **trinr4** curve could be considered to be positioned close to the other 5 curves. Figure 4.38 shows that the multifractality of the resultant Laplacian fractals is relatively unaffected by the variation of the parameters listed in Table 4.1.

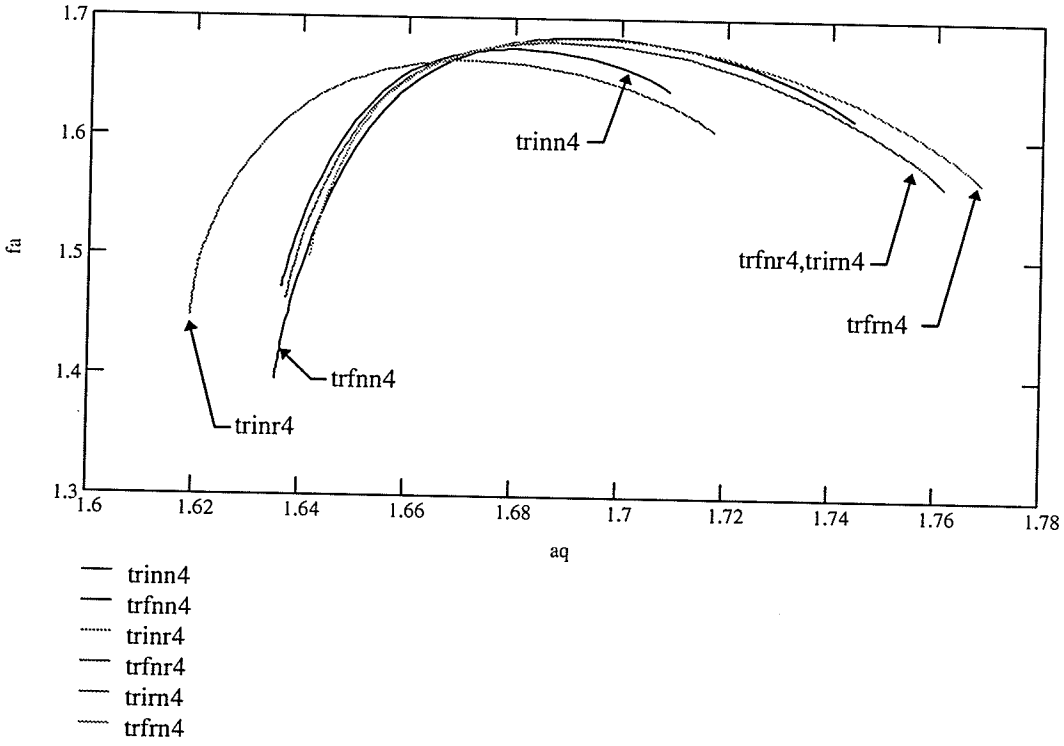


Fig. 4.38: Effect of parameters on the Mandelbrot dimension.

4.9.2 Experiment 8: Effect of Random and Chaotic Generators

The purpose of this experiment was to assess the impact of the random or chaotic generator on the resultant structure and multifractality of the Laplacian fractals. Only the generator was varied between runs. The Mandelbrot dimensions are shown for 8 pseudo-random generators (Fig. 4.39) and 8 chaotic generators (Fig. 4.40). The generated Laplacian fractals and the final field patterns appear in Appendix E. The values for the morphological dimensions appear in Table 4.8, in ascending order according to D_0 (D_S).

Table 4.8: Morphological dimensions for different random and chaotic generators.

Generator	Filename	D_0, D_S	D_1, D_I	D_2, D_C
Rössler	trfnn14	1.136	1.131	1.127
Julia	trfnn12	1.480	1.472	1.464
Lorenz	trfnn13	1.560	1.558	1.555
Hénon	trfnn8	1.642	1.638	1.635
rand0	trfnn1	1.664	1.658	1.654
rand3	trfnn5	1.667	1.663	1.659
randq1	trfnn6	1.672	1.665	1.659
rand	trfnn0	1.675	1.669	1.665
Gingerbreadman	trfnn10	1.677	1.665	1.656
randq2	trfnn7	1.678	1.675	1.673
rand2(81)	trfnn4	1.684	1.677	1.672
rand1	trfnn2	1.687	1.678	1.672
rand2	trfnn3	1.690	1.682	1.677
Gaussian	trfnn15	1.721	1.716	1.712
Ikeda	trfnn11	1.752	1.746	1.742
Lozi	trfnn9	1.759	1.756	1.753

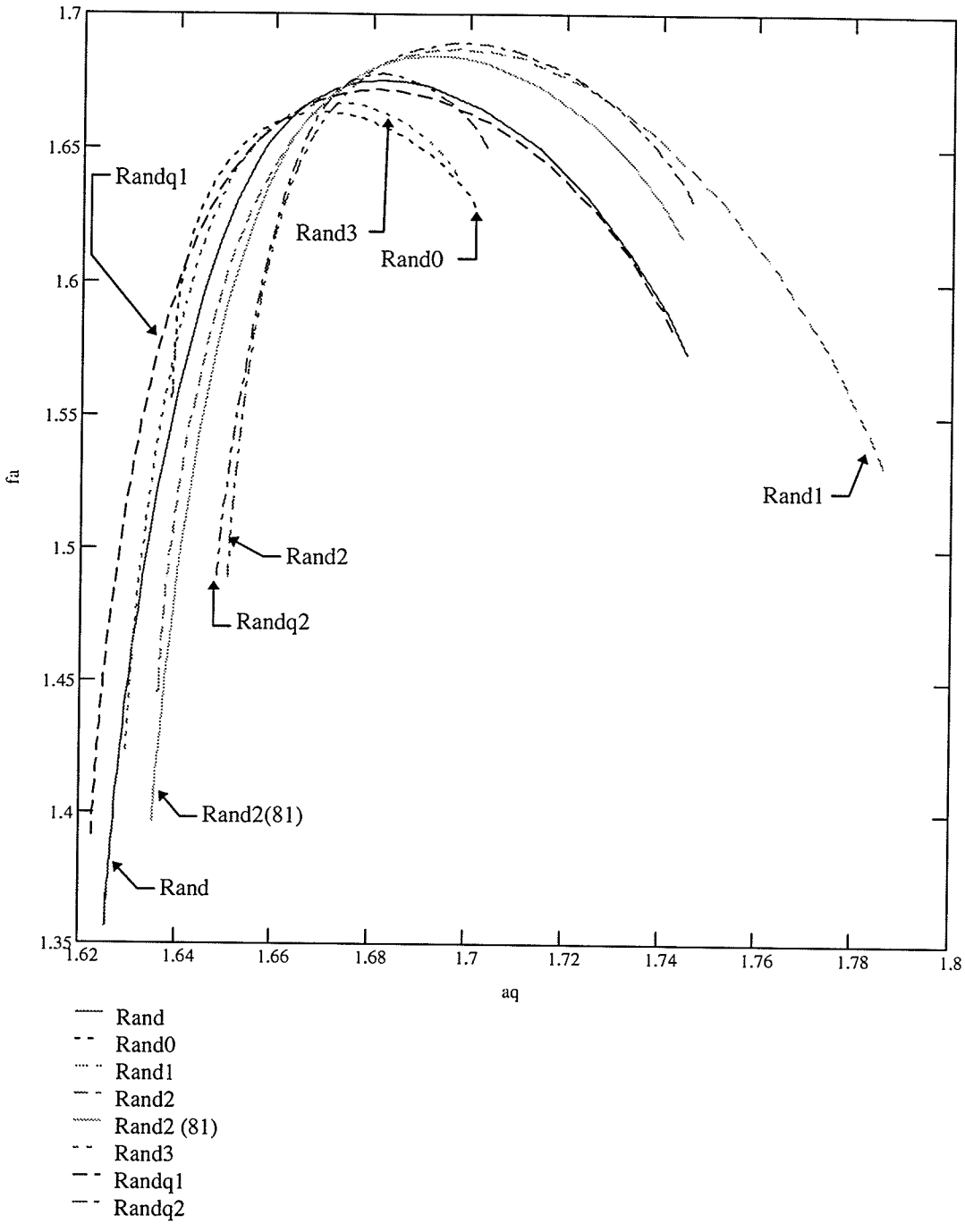


Fig. 4.39: Effect of random generators on Mandelbrot dimension.

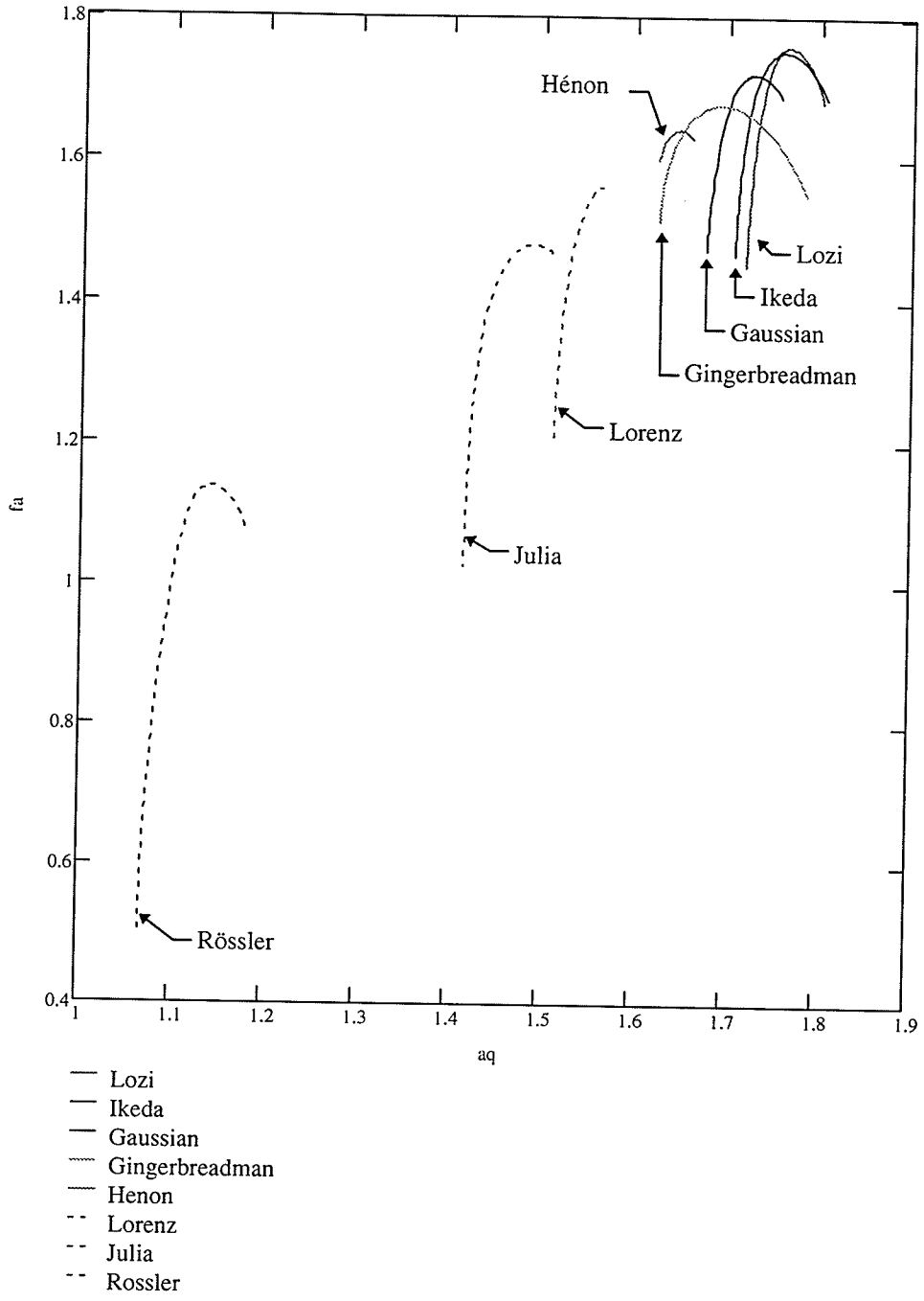


Fig. 4.40: Effect of chaotic generators on Mandelbrot dimension.

There are several conclusions that can be made concerning these results.

- The nature of the underlying process which causes the random growth process, whether it is due to dielectric material imperfections or other factors, here represented by the selected chaotic or pseudo-random generator, has a severe impact on the multifractality of the resultant structures. Generators which have similar properties will facilitate the growth of similar discharge structures. The Mandelbrot dimension curves of the 8 simulations which employed pseudo-random generators exhibited a very tight grouping, while the Mandelbrot dimension curves of the 8 simulations which employed chaotic (or the Gaussian) generator exhibited a high degree of separation from each other. Only the simulation using the Gingerbreadman generator occupied the same dimension space as the pseudo-random generators. It should be noted that the chaotic generators normalize the probability density function of the growth probabilities differently than the uniform (pseudo-random) generators.
- For chaotic generators which exhibit a high persistence, as measured by the variance dimension, and categorized as either brown or black noise [App. B], the resultant structure depends highly on the tag array scanning order. Notice that the simulation using the Rössler chaotic generator, which has the same variance dimension as black noise, the discharge heads into the oncoming left-to-right column scanning.
- There appear to be some systematic errors in the computation of the generalized correlation dimensions which are not monotonically decreasing as predicted by theory (Sec. 2.5.6). This phenomena (Fig. 4.41) has been reported elsewhere [FeKi95], and is beyond the scope of this thesis. It does, however, cause the Mandelbrot dimension curves to become multi-valued curves (Fig. 4.42). To rectify this, the range of q for

each Mandelbrot dimension curve is restricted to those q for which it becomes a single-valued function.

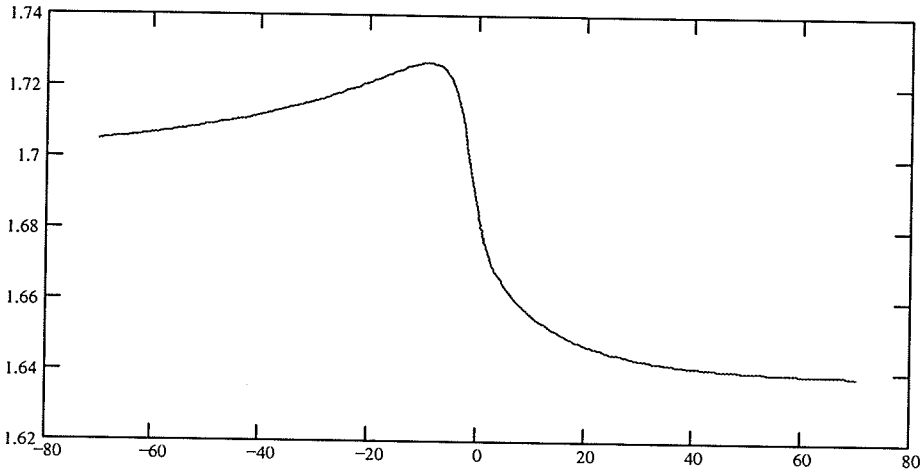


Fig. 4.41: Typical non-monotonicity in computed Rényi dimensions.

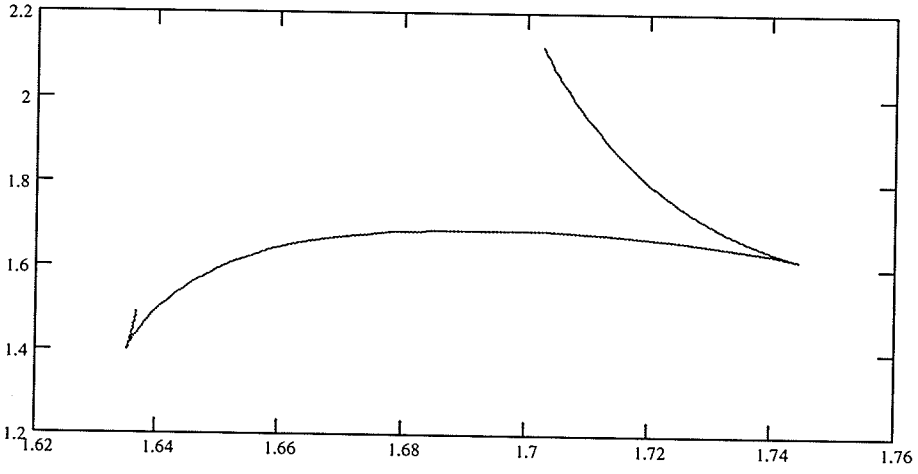


Fig. 4.42: Effect of non-monotonic Rényi dimensions on Mandelbrot dimension.

4.9.3 Experiment 9: Effect of Iteration Methods

The purpose of this experiment was to assess the impact of the iteration method on the resultant structure and multifractality of the Laplacian fractals. The Mandelbrot

dimensions are shown for several methods. Prior to the implementation of dynamic acceleration, simulations were performed to test the 16 iterative methods, however, accelerated methods used an acceleration factor of ω=1.0. In effect, the accelerated methods were equivalent to their Gauss-Seidel counterparts. For multifractal comparative purposes, only the simulation results of Gauss-Seidel based methods and the dynamically accelerated methods (SOR, SLOR, ADSLOR, and SBOR) are presented here. The generated Laplacian fractals and the final field patterns appear in Appendix E. The values for the morphological dimensions appear in Table 4.9.

Table 4.9: Morphological dimensions for iteration methods.

Filename	global updates	local updates	D_0, D_S	D_1, D_I	D_2, D_C
app220	GS	GS	1.681	1.674	1.669
app231	SOR	GS	1.671	1.665	1.661
app230	SOR	GS	1.672	1.668	1.661
app330	SOR	SOR	1.667	1.660	1.655
treem06	GLOR	GS	1.677	1.669	1.664
treem07	ADGLOR	GS	1.670	1.660	1.654
app280	SLOR	GS	1.670	1.662	1.656
treem12	GBOR	GS	1.676	1.667	1.661
app2d0	SBOR	GS	1.636	1.625	1.617

From Table 4.9 and Fig. 4.43, it can be seen that the selection of an iterative method to solve the potential field has very little effect on the computed morphological dimensions and the multifractal dimension, resulting in a tight grouping of the Mandelbrot dimensions.

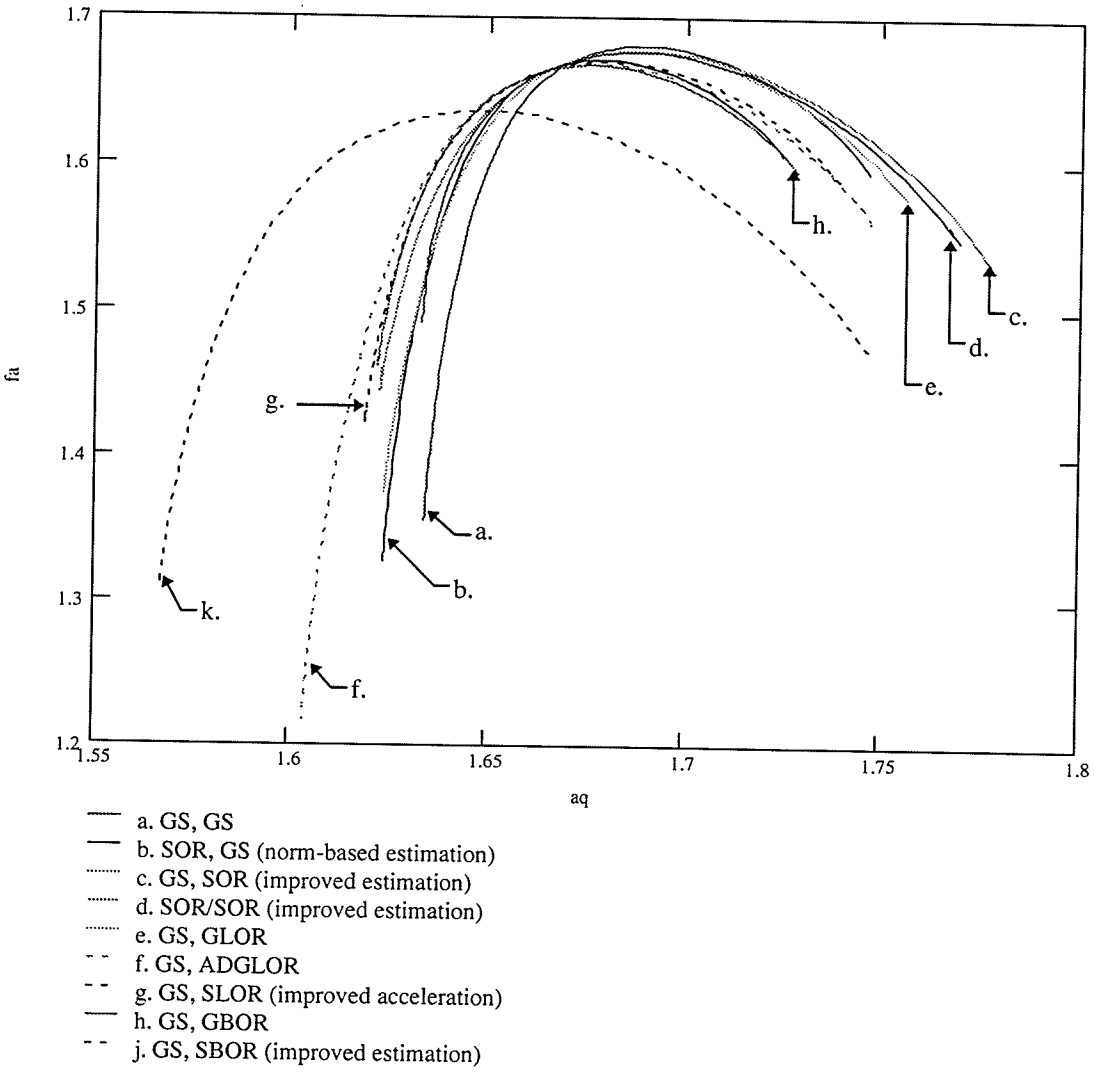


Fig. 4.43: Effect of iterative methods on Mandelbrot dimension.

The *only exception* to this is the simulation employing the accelerated SBOR iterative method. It was stated previously (Sec. 4.6.4) that this simulation yielded an incorrect potential field along the Neumann boundaries, due to incorrect coding. It is interesting to note that this problem separates the resulting Mandelbrot dimension, or fingerprint, from the other fingerprints. Thus, it may be possible to use multifractal measures to indicate which simulations are incorrect or are experiencing some sort of numerical problems.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1. Iterative Methods

The evaluation of the various iterative methods, focusing on their computational performance and their suitability for utilization to iteratively solve the mesh potentials as required by the dielectric breakdown model, has revealed many challenges and limitations.

The classical formulas tend to address only singularity-free, square mesh regions. While evaluating the initial results of the analysis of convergence, it was discovered that the theoretical values of ω_{opt} predicted by the classical formulas for SBOR and SLOR differed grossly from the experimental results, however estimates for ω_{opt} predicted by formulas published by Evans [Evan62, Evan64b] were within 2% of the experimental results.

The classical formulas methods for SLOR and ADSLOR were adapted to handle rectangular regions. Evaluation of experimental and predicted values of the optimal acceleration factor indicated that further tuning was required to avoid over-accelerating the iterative process into instability.

With regards to the convergence time, accelerated 2×2 block (SBOR) methods appear to have a slight convergence time advantage over accelerated point methods, however, SBOR would be less suitable than SOR to the extension of non-homogeneous and non-linear dielectrics. Line methods require the longest time for convergence due to

the computational workload of the tridiagonal inverse required for each line. Simulations using line methods took about 6 times longer than simulations using point methods. Line methods are *not* recommended for further evaluation.

Due to the improved convergence offered by accelerated methods for even poor estimates of the spectral radius λ_{max} of the iterative process, non-accelerated methods based on Gauss-Seidel (successive displacements) and especially Jacobi (simultaneous displacements) iterative methods are no longer recommended.

When employing accelerated methods to solve for the potential field, the final acceleration factor does not differ dramatically from the initial acceleration factor which suggests that even a somewhat sloppy *fixed* underestimate of ω_{opt} could be employed, obviating the need to use norm-based *dynamic* estimates of ω_{opt} and the associated mesh re-scanning time. Alternatively, a better, near-optimal estimate of ω_{opt} based on the maximum spectral radius of a collection of singularity-free rectangular mesh regions, and superior to norm-based methods was developed and used, improving the error reduction.

5.2. Multifractality

Previous research by [Aren94, Stac95, KiSt94] indicated that the Laplacian fractals generated by the discharge model were in fact multifractals. Analysis of the generated fractals using the Mandelbrot dimension showed that these objects were indeed multifractals. Due to a phenomenon which is not understood, and which is beyond the scope of this thesis, the Rényi dimension curves have a bump near q = -4, and are monotonically decreasing only in the range of $-4 \leq q \leq +\infty$. Utilization of f_α ands α_q information corresponding to $-\infty \leq q \leq -4$ results in a nonsensical, double-valued function

for the Mandelbrot dimension. This effect has also been noted elsewhere [FeKi95, Chen95] and is believed to be a systematic error. The negative impact of this is that the right side of most Mandelbrot dimension curves are much sorter than the left side, resulting in the loss of valuable information.

In measuring the multifractality of the resultant Laplacian fractals, it was found that the selection of the random generator had the greatest impact on the multifractality, causing the self-similarity dimension, D_S, to vary from 1.127 to 1.753. The impact of changing scanning directions, potential representation, and the iteration method and acceleration methods, is surprisingly minimal. If only 2 significant digits are considered, then $D_S = D_I = D_C = 1.6$! If 3 significant digits are considered, then

- $1.636 \leq D_S = D_0 \leq 1.68$,
- $1.625 \leq D_I = D_1 \leq 1.68$,
- $1.617 \leq D_C = D_2 \leq 1.67$.

Even for simulations employing acceleration, the morphological dimensions suffered no major side effects.

5.3. Summary

The major contributions of this thesis are:

- the extension of line and block iterative methods to operate in the presence of singularities in the interior of a discrete mesh,
- the development of a new and improved method of estimating λ_{max} and ω_{opt} in the event of non-stationary boundaries (singularities) in the mesh interior, and

- the characterization of the multifractality of Laplacian discharges using the Rényi and Mandelbrot dimensions,
- the evaluation of the impact of:
    - ◊ the representation (floating point or integer) of the mesh potentials,
    - ◊ the selected iterative method selected to solve for the potentials, and
    - ◊ the random generator employed

    on the multifractality of the resultant Laplacian fractals.

Since this thesis was completed within the constraints of finite available time, further research in particular areas is recommended. It is recommended that the effects of accelerated iterative methods on topologies other than parallel plates, such as coaxial topologies, be evaluated. Additionally, the extension and suitability of estimating λ_{max} and hence the estimation of ω_{opt} from singularity-free regions should be examined due to the reduced singularity-free region sizes caused by the curvature of the outer electrode intruding into the mesh interior, which will result in poorer estimates of ω_{opt}. The effect of including a breakdown voltage threshold (dielectric withstand voltage) in the dielectric breakdown model remains to be determined.

Finally, the extension of the dielectric breakdown model and the associated iterative methods to handle non-homogeneous and non-linear dielectrics is required before this technology can be applied towards the improved analysis and synthesis of realistic dielectric discharges and towards the improved design of dielectrics to *prevent* the occurrence of destructive dielectric discharges.

REFERENCES

[Acto70] F.S. Acton, *Numerical Methods That [Usually] Work*, Harper & Row, New York, 1970, 541 pp.

[AMSMP88] A.M. Albano, J. Muench, C. Schwartz, A.I. Mees, P.E. Rapp, "Singular-value decomposition and the Grassberger-Procaccia algorithm", *Physical Review A*, 38, no. 6, pp. 3017-3026, Sept. 15, 1988.

[Ames77] W.F. Ames, *Numerical Methods For Partial Differential Equations*, 2nd ed. Academic Press Inc., New York, 1977, 365 pp.

[Ames92] W.F. Ames, *Numerical Methods For Partial Differential Equations*, 3rd ed., Academic Press Inc., New York, 1992, 451 pp.

[Aren94] L.H. Arendt, "The effect of computational and modelling techniques on the fractal dimensions of dielectric discharges modelled using Laplacian fractals", *24.721: Fractals and Chaos Engineering* course project, Dept. of Electrical and Computer Eng., University of Manitoba, Winnipeg, Canada, June, 1994, 41 pp.

[Arend96] L. H. Arendt, "Stochastic Modeling and Multifractal Analysis of Dielectric Discharges Using Laplacing Fractals: Code Listings", *Technical Report, DEL96-10, v1.0*, Dept. of Electrical and Computer Eng., University of Manitoba, Winnipeg, Canada, June 28, 1996, 467 pp.

[ArGZ56] R.J. Arms, L.D. Gates and B. Zondek, "A method of block iteration", *J. Soc. Ind. Appl. Math. (J SIAM)*, **4**, No. 4, pp. 220-229, December 1956.

[AtSV88] H. Atmanspacher, H. Scheingraber, and W. Voges "Global scaling properties of a chaotic attractor reconstructed from experimental data", *Physical Review A*, **37**, No. 4, pp. 1314-1322, February 15, 1988.

[BaVi90] A-L Barabasi and Tamas Viscek, "Tracing a diffusion-limited aggregate: Self-affine versus self-similar scaling", *Physical Review A*, **41**, no. 12, pp. 6881-6883, 15 June 1990.

[BeCr73] G.E. Bell and J. Crank, "A method of treating boundary singularities in time dependant problems", *J. Inst. Maths Applics*, **12**, pp. 37-48, 1973.

[Bick48] W.G. Bickley, "Finite difference formulae for the square lattice", *Q. J. Mech. Appl. Math.*, **1**, pp. 8, 1948.

[BiGu74] G. Birkhoff and Surender Gulati, "Optimal few-point discretizations of linear source problems", *SIAM J. Numer. Analysis*, **11**, No. 4, pp. 700-727, September 1974.

[Birk71] G. Birkhoff, *The Numerical Solution of Elliptic Equations*, Harvard University, SIAM, Philadelpia, 1971, 82 pp.

[Brau57] A. Brauer, "A method for computing the greatest root of a positive matrix", *J. Soc. Ind. Appl. Math. (J SIAM)*, **4**, No. 4, pp. 250-253, December 1957.

[BSSW83] A. Brandstäter, J. Swift, H.L. Swinney, A. Wolf et all, "Low-dimensional chaos in a hydrodynamic system", *Physical Review Letters*, **51**, No. 16, pp. 1442-1445, 17 October 1983.

[BuSt66] R. Bulirsch and J. Stoer, "Numerical treatment of ordinary differential equations by extrapolation methods", *Numerische Mathematik*, **8**, pp. 1-13, 1966.

[CaLW69] B. Carnahan, H.A. Luther and J. Wilkes, *Applied Numerical Methods*, 1969, 604 pp.

[Carr61] B.A. Carré, "The determination of the optimum accelerating factor for successive over-relaxation", *Computer Journal*, **4**, pp. 73-78, 1961.

[Chen95] Hongjin Chen, "A Study of Computing Generalized Dimensions", *24.721: Fractals and Chaos Engineering* course project, Dept. of Electrical and Computer Eng., University of Manitoba, Winnipeg, Canada, April 19, 1995, 7 pp.

[ChKi95] Hongjin Chen and W. Kinsner, "A Study of Computing Generalized Dimensions", Dept. of Electrical and Computer Eng., University of Manitoba, Winnipeg, Canada, May 1995, 7 pp.

[Coll60] Collatz, *The Numerical Treatment of Differential Equations*, 3rd ed., P.G. Williams (trans.), NewYork: Springer-Verlag, 1960, 568 pp.

[CuVa59] E.H. Cuthill and R.S. Varga, "A method of normalized block iteration", *J.A.C.M.*, **6**, pp. 236-244, 1959.

[DeKi73a] E. Della Torre and W. Kinsner, "Solution to waveguide problems by successive extrapolated relaxation", *IEEE Trans. Microwave Theory Tech. (Short Papers)*, **MIT-21**, No. 4, pp. 490-491, July 1973.

[DeKi73b] E. Della Torre and W. Kinsner, "Convergence properties of the successive extrapolated relaxation (S.E.R.) method", *J. Inst. Maths Applics*, **12**, pp. 175-185, 1973.

[DeKi79] E. Della Torre and W. Kinsner, "The finite difference method with graded lattices", in *Perspectives on the Solution of Simultaneous Solutions*, A. Wexler (ed.), pp. 363-374, 1979.

[EcPr86] J-P Eckmann and I. Procaccia, "Fluctuations of dynamic scaling indicies in nonlinear systems", *Physical Review A*, **34**, No. 1, pp. 659-661, July 1986.

[Ehrl81] L.W. Ehrlich, "An ad hoc SOR method", in *Elliptic Problem Solvers*, Academic Press Inc., London, 1981, pp. 255-259.

[Evan62] D.J. Evans, "Note on the line over-relaxation factor for small mesh size", *Computer Journal*, **5**, pp. 48-50, 1962.

[Evan63] D.J. Evans, "The extrapolated modified Aitken iteration method for solving elliptic difference equations", *Computer Journal*, **6**, pp. 193-201, 1963.

[Evan64a] D.J. Evans, "The extrapolated modified Aitken iteration method applied to σ_1-ordered systems of linear equations", *Computer Journal*, **7**, pp. 137-140, 1964.

[Evan64b] D.J. Evans, "Estimation of the line over-relaxation factor and convergence rates of an alternating direction line over-relaxation technique", *Computer Journal*, **7**, pp. 318-321, 1964.

[Evan81] D.J. Evans, "On preconditioned iterative methods for elliptic partial differential equations", in *Elliptic Problem Solvers*, Academic Press Inc., London, 1981, pp. 261-269.

[Evan84] D.J. Evans, "Implicit block explicit overrelaxation schemes", in *Elliptic Problem Solvers II*, Academic Press Inc., London, 1984, pp. 285-300.

[Ever90] C Evertsz, "Self-affine nature of dielectric-breakdown model clusters in a cylinder", *Physical Review A*, **41**, no. 4, pp. 1830-1841, 15 Feb 1990.

[EvFo62] D.J. Evans and C.V.D. Forrington, "Note on the solution of certain tri-diagonal systems of linear equations", *Computer Journal*, **5**, 1963, pp. 327-328, 1962.

[Fabe81] V. Faber, "Block relaxation strategies", in *Elliptic Problem Solvers*, Academic Press Inc., London, 1981, pp. 271-275.

[Falc90] K.J. Falconer, *Fractal Geometry: Mathematical Foundations and Applications*, Chichester, New York: John Wiley & Sons Ltd., 1990, 288 pp.

[FaMi66] G. Fairweather and A.R. Mitchell, "Some computational results of an improved A.D.I. method for the Dirichlet problem", *Computer Journal*, **9**, pp. 298-303, 1966.

[FeKi95] K. Ferens and W. Kinsner, "Multifractal texture classification of images", *Proc. IEEE WESCANEX '95*, (Winnipeg, MB; May 15-16, 1993), IEEE Cat. No. 95CH3581-6, pp. 438-444, 1995.

[FoWa67] G.E. Forsythe and W.R. Wasow, *Finite-Difference Methods for Partial Differential Equations*, New York: John Wiley & Sons Inc., 1967, 444 pp.

[FrSw86] A.M. Fraser and H.L. Swinney, "Independant coordinates for strange attractors from mutual information", *Physical Review A*, **33**, No. 2, pp. 1134-1140, February 1986.

[FuSa87] J. Fukai and N. Sano, "Tree growth simulation associated with dielectric breakdown", *IEEE Proc. Conf. Electrical Insulation & Dielectric Phenomena*, pp. 432-439, 1987.

[Gera78] C.F. Gerald, *Numerical Solution of Elliptic Partial-Differential Equations*, Addison-Wesley Publishing Co., 1978, 518 pp.

[GeWh84] C.F. Gerald and P.O. Wheatley, *Applied Numerical Analysis*, 3rd ed., Addison-Wesley Publishing Co., 1984, 579 pp.

[GJLS86] J.A. Glazier, M.H. Jensen, A. Libchaber, and J. Stavans, "Structure of Arnold tongues and the $f(\alpha)$ spectrum for period doubling: Experimental results", *Physical Review A (Rapid Communications)*, **34**, No. 2, pp. 1621-1624, August 1986.

[Grass83a] P. Grassberger, "On the fractal dimension of the Henon attractor", *Physics Letters*, **97A**, no. 6, pp. 224-226, 5 September 1983.

[Grass83b] P. Grassberger, "Generalized dimensions of strange attractors", *Physics Letters*, **97A**, no. 6, pp. 227-230, 5 September 1983.

[Grass90] P. Grassberger, "An optimized box-assisted algorithm for fractal dimension", *Physical Letters A*, 148, no. 1,2, pp. 63-69, Aug. 6, 1990.

[GrKi95] W. Grieder and W. Kinsner, "Speech segmentation by variance fractal dimension", University of Manitoba, Winnipeg, 5 pp. [publication status unknown]

[GrPr83a] P. Grassberger and I. Procaccia, "Characterization of strange attractors", *Physical Review Letters*, **50**, No. 5, pp. 346-349, 31 January 1983.

[GrPr83b] P. Grassberger and I. Procaccia, "Estimation of the Kolmogorov entropy from a chaotic signal", *Physical Review A (Rapid Communications)*, **28**, No. 4,

pp. 2591-2593, October 1983.

[GrPr83c] P. Grassberger and I. Procaccia, "Measuring the strangeness of strange attractors", *Physica* **9**D, pp. 189-208, 1983.

[GrPr84] P. Grassberger and I. Procaccia, "Dimensions and entropies of strange attractors from a fluctuating dynamic approach", *Physica* **13**D, pp. 34-54, 1984.

[GWSP82] H.S. Greenside, A. Wolf, J. Swift, T. Pignataro, "Impracticality of a box-counting algorithm for calculating the dimensionality of strange attractors", *Physical Review A (Rapid Communications)*, **25**, No. 6, pp. 3453-3456, June 1982.

[Hadj68] A. Hadjidimos, "On a generalised alternating direction implicit method for solving Laplace's equation", *Computer Journal*, **11**, pp. 324-328, 1968.

[Hadj71] A. Hadjidimos, "Optimum extrapolated ADI iterative difference schemes for the solution of Laplace's equations in three space variables", *Computer Journal*, **14**, pp. 179-183, 1971.

[HaJM85] S.M. Hammel, C.K.R.T. Jones, J.V.Moloney, "Global dynamical behavior of the optical field in a ring cavity", *J. Opt. Soc. Am.* **B**, Vol. **2**, No. 4, pp. 552-564, April 1985.

[HaMP86] T.C. Halsey, P. Meakin, I. Procaccia, "Scaling structure of the surface layer of diffusion-limited aggregates", *Physical Review Letters*, **56**, No. 8, pp. 854-857, 24 February 1986.

[Hayt81] William H. Hayt, Jr., *Engineering Electromagnetics*, 4th ed., McGraw-Hill Inc., Toronto, 1981, 527 pp.

[Hell60] J. Heller, "Simultaneous, successive, and alternating direction iteration schemes", *J. Soc. Ind. Appl. Math. (J SIAM)*, **8**, No. 1, pp. 150-173, March 1960.

[HePr83] H.G.E. Hentschel and I. Procaccia, "The infinite number of generalized dimensions of fractals and strange attractors", *Physica* **8**D, pp. 435-444, 1983.

[HJKPS86] T.C. Halsey, M.H. Jensen, L.P. Kadanoff, I. Procaccia, and B.I. Shraiman, "Fractal measures and their singularities: The characterization of strange sets", *Physical Review A*, **33**, No. 2, pp. 1141-1151, February 1986.

[HyMa84] J.M. Hyman and T.A. Manteuffel, "Dynamic acceleration of nonlinear iterations", in *Elliptic Problem Solvers II*, Academic Press Inc., London, 1984, pp. 301-313.

[KiDe74] W. Kinsner and E. Della Torre, "An iterative approach to the finite-element method in field problems", *IEEE Trans. Microwave Theory Tech.*, **MIT-22**, No. 3, pp. 221-228, March 1974.

[KiHa90] D.R. Kincaid and L.J. Hayes (eds.), *Iterative Methods for Large Linear Systems*, Boston: Academic Press Inc., 1990, 319 pp.

[Kins94a] W. Kinsner, "Self-similarity: fractals, chaos, scaling and their applications", *Technical Report DEL94-2, v2.43*, Dept. of Electrical and Computer Eng., University of Manitoba, Winnipeg, Canada, January 31, 1994, 113 pp.

[Kins94b] W. Kinsner, "Fractional Brownian noise and the variance fractal dimension", *Technical Report DEL94-3, v2.86*, Dept. of Electrical and Computer Eng., University of Manitoba, Winnipeg, Canada, April 15, 1994, 86 pp.

[Kins94c] W. Kinsner, "Entropy-based fractal dimensions: probability and pair correlation algorithms for E-dimensional images and strange attractors", *Technical Report DEL94-5, v3.31*, Dept. of Electrical and Computer Eng., University of Manitoba, Winnipeg, Canada, June 10, 1994, 44 pp.

[Kins94d] W. Kinsner, "Batch and real-time computation of a fractal fimension based on variance of a time series", *Technical Report DEL94-6, v3.27*, Dept. of Electrical and Computer Eng., University of Manitoba, Winnipeg, Canada, June 15, 1994, 22 pp.

[Kins94e] W. Kinsner, "Hausdorff-Besicovitch dimension", *Technical Report DEL94-7*, Dept. of Electrical and Computer Eng., University of Manitoba, Winnipeg, Canada, June 20, 1994, 12 pp.

[Kins94f] W. Kinsner, "Fractal & Chaos Engineering", course notes for *24.721: Fractals and Chaos Engineering*, Dept. of Electrical and Computer Eng., University of Manitoba, Winnipeg, Canada, 1994, 259++ pp.

[KiSt94] W. Kinsner and G. Stacey, "Computer modelling of Laplacian fractals in dielectric discharge phenomena", *Mathematical Modelling and Scientific Computing Journal*, 7 pp. (in press)

[KrCa73] John D. Kraus, Keith R Carver, *Electromagnetics*, 2nd ed., McGraw-Hil Inc., Toronto, 1973, 828 pp.

[Lewi71] R.P.W. Lewis, "The selection of ADI iteration parameters by numerical experiment for the solution of Poisson's equation over a circular area", *Computer Journal*, **14**, pp. 73-74, 1971.

[LlMc68] T. Lloyd and H. McCallion, "Bounds for the optimum over-relaxation factor for the S.O.R. solution of Laplace type equations over irregular regions", *Computer Journal*, **11**, pp. 329-331, 1968.

[LyRi78] R.E. Lynch and J.R. Rice, "High accuracy finite difference approximation to solutions of elliptic partial differential equations", *Proc. Natl. Acad. Sci. USA*, **75**, No. 6, pp. 2541-2544, June 1978.

[MaTe61] D.W. Martin and G.J. Tee, "Iterative Methods for Linear Equations with Symmetric Positive Definite Matrix", *Computer Journal*, **4**, pp. 242-254, 1961.

[McGu91] M. D. McGuire, *An Eye For Fractals: A Graphic/Photographic Essay*, Addison-Wesley Publishing Company Inc., Redwood City, 1991, 165 pp.

[MiPG84] A. Ben-Mizrachi, I. Procaccia, and P. Grassberger, "Characterization of experimental (noisy) strange attractors", *Physical Review A (Brief Reports)*, **29**, No. 2, pp. 975-977, February 1984.

[Motz47] H. Motz, "The treatment of singularities of partial differential equations by relaxation methods", *Q. Appl. Math.*, Vol. **IV**, No. 4, pp. 371-377, 1947.

[NiPW84] L. Niemeyer, L. Pietronero, and H.J. Wiesmann, "Fractal dimension of dielectric breakdown", *Physical Review Letters*, **52,** no. 12, pp. 1033-1036, 19 March 1984.

[Osbo64] M.R. Osborne, "A method for finite-difference approximation to ordinary differential equations", *Computer Journal*, **7**, pp. 58-65, 1964.

[Part81] S. Parter, "Block iterative methods", in *Elliptic Problem Solvers*, Academic Press Inc., London, 1981, pp. 375-382.

[PaSc87] K. Pawelzik and H.G. Schuster, "Generalized dimensions and entropies from a measured time series", *Physical Review A (Rapid Communications)*, **35**, No. 1, pp. 481-484, January 1, 1987.

[PeJS92] H.-O. Peitgen, H. Jurgens and D. Saupe. *Chaos and Fractals*, New York, NY: Springer-Verlag, 1992, 984 pp.

[PeSa88] H.-O. Peitgen and D. Saupe, *The Science of Fractal Images*, New York, NY: Springer-Verlag, 1988, 312 pp.

[PiEW86] L. Pietronero, C. Evertsz and H. J. Wiesmann, "Scaling properties of growing zone and capacity of Laplacian fractals", in *Fractals in Physics*, L. Pietronero and E. Tosatti (eds.), Elsevier Science Publishers, New York, NY, pp. 159-163, 1986.

[PrLi90] P. Prusinkiewicz and A. Lindenmayer, *The Algorithmic Beauty of Plants*, Springer-Verlag, New York, 1990, 228 pp.

[PTVF92] W. H. Press, S. A. Tuekolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, 2nd ed., New York, NY, Cambridge University Press, 1992, 994 pp.

[Rand67] T.J. Randall, "A note on the estimation of the optimum successive overrelaxation parameter for Laplace's equation", *Computer Journal*, **10**, pp. 400-401, 1967.

[Reid66] J.K. Reid, "A method for finding the optimum successive over-relaxation parameter", *Computer Journal*, **9**, pp. 200-204, 1966.

[RuHO80] D.A. Russell, J.D. Hanson, E. Ott, "Dimension of Strange Attractors", *Physical Review Letters*, **45**, No. 14, pp. 1175-1178, 6 October 1980.

[Satp86] S.Satpathy, "Dielectric breakdown in three dimensions", in *Fractals in Physics*, L. Pietronero and E. Tosatti (eds.), Elsevier Science Publishers, New York, NY: North Holland, pp. 173-176, 1986.

[ShWe38] G.H. Shortley and R. Weller, "The numerical solution of Laplace's equation", *J. Appl. Phys.*, **9**, pp 334-348, May 1938.

[Smit85] G.D. Smith, *Numerical Solution of Partial Differential Equations: Finite Difference Methods*, 3rd ed., Clarendon Press; New York: Oxford, 1985, 337 pp.

[Smit65] G.D. Smith, *Numerical Solution of Partial Differential Equations With Exercises and Worked Solutions*, London: Oxford University Press, 1965, 179 pp.

[SOYH82] Y. Sawada, S. Ohta, M. Yamazaki, and H. Honjo, "Self-similarity and a phase-transition-like behaviour of a random growing structure governed by a non-equilibrium parameter", *Physical Review A*, 26, no. 6, pp. 3557-3563, December 1982.

[Stac95] G. Stacey, *Stochastic Fractal Modelling of Dielectric Discharges*, M.Sc. Thesis, University of Manitoba, Winnipeg, Manitoba,1995, 298 pp.

[Stev90] R.T. Stevens, *Fractal Programming in Turbo Pascal*, California: M&T Pub., 1990, 462 pp.

[StKi95] G. Stacey and W. Kinsner, "Tuning of the stochastic model of dielectric discharges", *Proc. IEEE WESCANEX '95*, (Winnipeg, MB; May 15-16, 1993), IEEE Cat. No. 95CH3581-6, pp. 318-320, 1995.

[Varg62] R.S. Varga, *Matrix Iterative Analysis*, N.J.: Prentice-Hall, 1962, 322 pp.

[Varg84] R.S. Varga, "A survey of recent results", in *Elliptic Problem Solvers II*, Academic Press Inc., London, 1984, pp. 197-217.

[Vics92] T. Vicsek. *Fractal Growth Phenomena.* (2nd ed.), Singapore: World Scientific, 1992, 488 pp.

[Waso55] W. Wasow, "Discrete approximations to elliptic differential equations", *ZAMP*, Vol. **VI**, pp. 81-97, 1955.

[Wilk65] J.H. Wilkinson, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.

[WiPi86] H. J. Wiesmann and L. Pietronero, "Properties of Laplacian fractals for Dielectric breakdown in 2 and 3 dimensions", in *Fractals in Physics*, L. Pietronero and E. Tosatti (eds.), Elsevier Science Publishers, New York, NY, pp. 151-157, 1986.

[WiSa81] T. A. Witten, Jr. and L. M. Sander, "Diffusion-limited aggregation, a kinetic critical phenomenon", *Physical Review Letters,* **47**, No. 19, pp. 1400-1403, 9 November 1981.

[YaHu93] Y. Yamamoto and R.L. Hughson, "Extracting fractal components from time series", *Physica D*, **68**, pp. 250-264, 1993.

[Young54] David M. Young, "Iterative methods for solving partial differential equations of elliptic type", *Trans. Amer. Math. Soc.*, **76**, p. 92-111, 1954.

[Young71] David M. Young, *Iterative Solution of Large Linear Systems*, New York: Academic Press Inc., 1971, 570 pp.

[YoGr73] David. M. Young, T.D. Gregory, *A Survey of Numerical Mathematics*, (2 vols), Reading, Mass.: Addison-Wesley Publishing Company, 1973.

[YoKi81] David M. Young and D.R. Kincaid, "The ITPACK package for large sparse linear systems", in *Elliptic Problem Solvers*, Academic Press Inc., London, 1981, pp. 163-184.

APPENDIX A

TIMING OF SIMULATION RUNS

A.1. Introduction

This appendix contains the simulation times for several simulation runs performed for this thesis. As all the simulations involve intensive computations with very little overhead required by the screen updates, these run times are indicative of the numerical effort required to simulate the dielectric breakdown phenomenon.

A.2. Hardware Utilized for Experiments

The timing information is compiled from runs performed on one of two personal computers configured as shown in Table A.1. To obtain the timing data for evaluating the iterative methods, the 486 machine was used as its slower execution speed, and hence longer execution times, allowed more precise time measurements than the fast Pentium permitted. The simulations of the Laplacian fractals were performed exclusively on the Pentium machine, as were the computations of the generalized correlation dimensions. All times are approximate, and not all simulations were timed.

Table A.1: Experimental Hardware.

Processor	Processor speed	Installed DRAM	Operating System
486DX-33	33 MHz	8 MB	Windows 3.1
Pentium-150	150 MHz	32 MB	Windows 95

A.3. Simulation Times

Table A.2: Simulation times.

Experiment	Comments	Machine	Simulation time
1	effects of model variations	Pentium-150	5.0 to 6.4 hrs.
2	timing of execution of methods	486DX-33	5.3 hrs.
3,4,5	iterations until convergence	Pentium-150	> 9.5 hrs.
6	eval. of norm-based estimates	486DX-33	92.5 s/pnt/pass
6	eval. of norm-based estimates	Pentium-150	12.85 s/pnt/pass
7,8	effects of various conditions	Pentium-150	7.0 to 13.5 hrs.
9	point methods	Pentium-150	6.5 to 11.0 hrs.
9	line methods	Pentium-150	35.0 to 48.0 hrs.
9	block methods	Pentium-150	6.5 to 11.0 hrs.
N/A	comp. of correlation dimension	Pentium-150	25 m.

The data for experiments 3,4, and 5 were obtained from a *single* simulation run. Based on timing results of experiment 6, a simple calculation shows that the Pentium is 7.194 times faster than the 486 for our application.

APPENDIX B

RANDOM AND CHAOTIC GENERATORS

B.1. Introduction

In order to implement the stochastic nature of the dielectric breakdown model, a method of *randomly* selecting a growth site is required. This requires the use of some random variable $R(t)$. Since the growth probability of a given site should ideally only be related to the enhanced electric field (Eq. 2-12), uniform deviates should be used. Uniform deviates are a set R of N_R random numbers, lying within some specified range, with each random number R_j having the same probability p_j of selection such that $p_j = 1/N_R$.

It is impossible to generate ideal uniform deviates, so the focus of this section is the analysis and characterization of a selection of pseudo-random and chaotic generators. The suitability of these generators for purposes other than the dielectric breakdown model is beyond the scope of this thesis.

B.2. Pseudo-Random Number Generators

Pseudo-random generators (PRNGs) will by definition generate a repeating, finite-length sequence of random numbers. The sequence, however long it may be, will eventually repeat itself, and the generator should be selected so that its minimal sequence is much longer than number of random points actually required. Most PRNGs are based on the linear congruential method [PTVF92]. Pseudo-random generators are generally

fast, easy to calculate, and have standard portable implementations. Quite often, however, pseudo-random number generators are not free from sequential correlation, may suffer from period exhaustion, and may have a limited range of values (16-bit generators).

The reader is asked to refer to the 1988 and 1992 [PTVF92] editions of *Numerical Recipes in C* for additional information on the different pseudo-random generators described below.

B.2.1 rand()

This is the standard *rand* function provided by all C and C++ compilers. It is characterized by a small dynamic range of values, sequential correlation, and period exhaustion. It is included here for comparative purposes.

B.2.2 Rand0

This generator is considered to be the minimal standard. The period of *Rand0* is about 2×10^9. It suffers from a limited resolution and serial correlations [PTVF92].

B.2.3 Rand1

Rand1 implements a Bays-Durham shuffle on the generated values to reduce the low-order serial correlations present in *Rand0*. It has a period of about 10^8.

B.2.4 Rand2

Rand2 uses multiple linear congruential generators with output-shuffling to yield a period near 2×10^{18}, and to reduce the low-order serial correlations in *Rand0*.

B.2.5 Rand2_81

Rand2_81 is the *Rand2* algorithm found in the 1981 edition of *Numerical Recipes in C*. It was utilized by Stacey in his research and is included here so that results can be duplicated and compared.

B.2.6 Rand3

Rand3 is a portable routine based on a subtractive method.

B.2.7 Randq1 and Randq2

These two fast generators are referred to as "quick and dirty generators" used to somewhat randomize a given system. They are computationally fast, with *Randq2* being machine and platform-dependent [PTVF92], and is therefore not recommended for general use in programs.

B.2.8 Gaussian Generator

A random variable G having a normal Gaussian distribution with a mean μ=0 and a variance $\sigma^2 = 1$, is generated by iterating

$$G_{n+1} = 2\left(R_n + S_n + T_n\right) - 3 \qquad \text{(B-1)}$$

where R, S, and T are uniform random variables, each with a mean μ and variance σ^2 of

$$\mu = 0.5, \quad \sigma^2 = {}^{1}\!/_{12}, \quad \text{for} \quad 0 \le R, S, T < 1.0 \qquad \text{(B-2)}$$

Therefore the variance of G is

$$\sigma_G = \sigma^2_{2(R+S+T)-3} = 4\sigma^2_{R+S+T} = 4\left(\sigma_R^2 + \sigma_S^2 + \sigma_T^2\right) = 1.0 \qquad \text{(B-3)}$$

and the mean of G is

$$\mu_G = \mu_{2(R+S+T)-3} = 2\mu_{R+S+T} - 3 = 2(3)(0.5) - 3 = 0.0 \qquad \text{(B-4)}$$

as expected for a normal distribution. Alternatively, if n uniform random variables are used, then

$$G_{n+1} = \sqrt{\frac{12}{n}} \sum_{i=1}^{n} R_i - \sqrt{3n} \qquad \text{(B-5)}$$

If n=3, then we recover Eq. B-1, and avoid the calculation of square roots.

B.3. Chaotic Generators

The patterns or structures that arise from dissipative dynamical systems, or from iterating linear transformations as the iteration index n tends to infinity, are called strange attractors [PeJS92, p. 655]. Iterations of very simple formulae can yield very beautiful and complex attractors having fractal, and possibly multifractal, properties.

One implied property of strange attractors is that they are self-avoiding, therefore all points on the attractor are unique (as are each *set* of point coordinates), regardless of how closely the points may be clustered. [The exceptions are the stable points of the attractor, which are not attained unless the iterations start at a stable point, in which case they will also remain there indefinitely.] This suggests the use of the coordinates of the attractor points as a source for an infinitely long sequence of unique numbers, in other words, the attractor is used as a *chaotic generator*.

The first difficulty lies in the dilemma that 2 numbers, x_i and y_i are used to describe each attractor point P_i: should the x_i and y_i values be used sequentially or combined in some fashion into one single number? For some attractors such as the Hénon, Lozi, and Gingerbread Man attractors, the y coordinate is linearly related to the x

coordinate, and the dilemma vanishes. Secondly, the uniqueness of *each set of point coordinates* (x_i, y_i) does not guarantee the uniqueness of all *values* of x_i or of all *values* of y_i. Alternatively, the uniqueness property is not guaranteed for projections of the fractal attractor, which in our case is onto the x or y axis.

Several well known attractors were selected for use as chaotic generators, with the x coordinate used as the random variable $R(t)$. Tests were performed to check for the occurrence and frequency of duplicate values and duplicate sequences within the first 100,000 values for the x and y coordinates. This ascertained if the function $R(t)$, consists of a unique sequence with unique values for $0 \leq t \leq 100{,}000$.

Furthermore, it is desired to normalize $R(t)$ to a range of $0 \leq R(t) \leq 1.0$ so it can be scaled in Eqs. 2-12 and 2-13. The extents of the attractors as $n \rightarrow \infty$ are unknown and intractable. For simulations with 510×510 grids, less than 30,000 random values are required, so the extents of the first 100,000 values are used.

The Hénon, Lozi and Gingerbread Man attractors are obtained by simple, related iterative formulas in 2-dimensional real space, the Julia and Ikeda attractors are obtained by iterations in the complex plane, and the Lorenz and Rössler attractors are obtained by iterations in 3-dimensional space. With the exception of the Ikeda attractor, the chaotic generators were just as simple if not simpler to implement than the shuffle-enhanced PRNGs. Plots of the strange attractors can be found in Sec. B.6.

B.3.1 Hénon Attractor

The Hénon attractor, Fig. B-1, [PeJS92] is generated by iterating

$$x_{n+1} = -ax_n^2 + 1 + y, \quad n = 0,1,2,\ldots$$
$$y_{n+1} = bx_n \tag{B-6}$$

for a=1.4 and b=0.3. It should be noted that some points on the attractor appear to lie very close together, and are indistinguishable using 32-bit **float**s for the x and y coordinates. These points *are* distinguishable when using 64-bit **double** [precision] variables for the x and y coordinates.

B.3.2 Lozi Attractor

The Lozi attractor, Fig. B-2, [PeJS92] is generated by iterating

$$x_{n+1} = -a|x_n| + 1 + y, \quad n = 0,1,2,\ldots$$
$$y_{n+1} = bx_n \tag{B-7}$$

for a=1.7 and b=0.5.

B.3.3 Gingerbread Man Attractor

The Gingerbread Man attractor, Fig. B-3, [PeSa88] is generated by iterating

$$x_{n+1} = |x_n| + 1 - y, \quad n = 0,1,2,\ldots$$
$$y_{n+1} = x_n \tag{B-8}$$

B.3.4 Ikeda Attractor

Another chaotic generator considered was the Ikeda attractor, Fig. B-4, generated by the plane-wave model of a bistable optical ring cavity proposed by Ikeda [HaJM85]. The attractor is generated by iterating in the complex plane

$$g_n = 0.85 + 0.90\exp\left[i\left(0.40 - \frac{5.5}{1 + \|g_{n-1}\|_2^2}\right)\right]g_{n-1} \tag{B-9}$$

B.3.5 Julia Attractor

Julia sets are obtained by iterating in the complex plane

$$z_{n+1} = z_n^2 + c, \quad n = 0,1,2,\ldots \tag{B-10}$$

The structure of Julia sets are dependent on the value of c, and can be classified [McGu91, p. 76] into:

- Cantor-like dust,
- separated clusters of points,
- "connected" points with no interior, or
- "connected" points with a solid interior.

Projections of Julia sets in these categories onto the x-axis result in disconnected points, disconnected line segments, or a connected line. Julia sets with solid interiors generally are thicker in the middle and tapering to a point at either horizontal extremes, yielding a bell-shaped distribution for discretized intervals of $x(t)$, and structures with separated clusters of points are characterized by isolated regions of bell-shaped probabilities. Since the objective is to use $x(t)$ as a random variable having a uniform distribution, a structure with "connected" points and without an interior, Fig. B-5, [c = -0.83+0.16i, PeJS92, p. 863] was selected, the idea being that if the connected points were not clustered, a fairly uniform distribution could be obtained. It turned out that they are instead characterized by a probability distribution function containing spikes many times higher than the average probability. This property can actually be utilized to model *localized defects in the dielectric material* and thereby enhance the randomness of the dielectric breakdown model. The iteration algorithm was implemented as per PeJS92.

B.3.6 Lorenz and Rössler Attractors

The Lorenz (Fig. B-6) and Rössler (Fig. B-7) attractors are based on continuous dynamical systems [Stev90, PeJS92]. Plots of the x or y coordinates of either attractor yield "slowly-changing", smooth, differentiable-almost-everywhere curves unlike the scattered dust for the $x(t)$ and $y(t)$ values for the Hénon, Lozi, and Gingerbread Man attractors. As a result, these two attractors were deemed unsuitable for use as chaotic generators in our system.

B.4. Characteristics

In comparing random and chaotic generators, essentially three properties are measured:

- the complexity of their generator or generating function,
- their probability distribution, which determines the likelihood of a value R_j being selected, and
- the randomness of their ordering, or sequencing of the values.

All measures of the random and chaotic generators are based on a sample space encompassing the first 100,000 points. The reader is asked to refer to the C++ code listings for the random and chaotic generators in the *srand.cpp*, *srand.h* and *sysrand.h* files in Appendix H. All the data for the graphs and tables that follow can be generated by running the *random.exe* program, the code for which appears in Appendix F.

B.4.1 Properties of an Ideal Uniform Distribution

If we consider an ideal uniform discrete random variable R which can equally assume N possible values on the unit interval $0 \le j \le 1$, such that $p_j = 1/N$, then it can be shown that its mean μ and variance σ^2 are defined as

$$\mu(R) = \frac{\sum_{j=0}^{N-1} \frac{1}{N}}{N} = \frac{1}{2}\left(1 - \frac{1}{N}\right) \tag{B-11}$$

$$\sigma^2(R) = \frac{1}{12}\left(1 + \frac{1}{N}\right) \tag{B-12}$$

The $x(t)$ and $y(t)$ of the selected attractors are not constrained to the unit interval, so for comparative and scaling purposes, all $x(t)$ are normalized to exist on the unit interval such that $0 \le x(t) \le 1$. The normalized function, $x_n(t)$ is defined as

$$x_n(t) = \frac{x(t) - x_{\min}}{x_{\max} - x_{\min}} \tag{B-13}$$

and has a mean μ_n and variance σ_n^2 defined in terms of $x(t)$, $\mu(x(t))$ and $\sigma^2(x(t))$ as

$$\mu_n(x_n(t)) = \frac{1}{N}\sum_{j=1}^{N}\left(\frac{x(t) - x_{\min}}{x_{\max} - x_{\min}}\right) = \frac{\mu(x(t)) - x_{\min}}{x_{\max} - x_{\min}} \tag{B-14}$$

$$\sigma_n^2(x_n(t)) = \frac{1}{(x_{\max} - x_{\min})^2}\sigma^2(x(t)) \tag{B-15}$$

B.4.2 Complexity of Chaotic Generators

The complexity of chaotic generators can be estimated from the topological dimension since their attractors are embedded in the x-y plane. It can be shown that the topological dimension of any projection of the attractor cannot exceed the dimension of the original attractor. The pseudo-random number generators produce a temporal signal

based on a single state variable, and how many are actually involved is unknown. The embedding and fractal dimensions of the pseudo-random generators are unknown but can be determined by Taken's method [AMSMP88, YaHu93], which was beyond the scope of this thesis.

The Hausdorff mesh counting dimension D_{HM} was used to measure the topological dimension of the strange attractors of each chaotic generator. For each attractor, a 100,000 points were generated. The horizontal and vertical extents of the attractor were used to create a bounding rectangle which was then subdivided into 512×512 equal-sized rectangular vels. The number of vels, N_k, intersected by the attractor at 1/512 scale were counted, then blocks of 2×2 block of vels were merged into larger vels, and counted as before but at 1/256 scale. This process was done for k=$\log_2(512)$ stages after which the attractor was covered by a single vel. The slope of the best fitting line of the $\log(N_k)$ vrs $\log(1/r)$ data is the value of D_{HM}.

The dimension plots for the attractors appear at the end of this appendix. The values for the dimensions appear in Tables B-2 and B-3. It should be noted the above procedure measures the fractal dimensions of the *2-dimensional projections* of the 3-dimensional Lorenz and Rössler attractors.

B.4.3 Measures of x(t) and y(t)

The measures of $x(t)$ are concerned with the probability distribution of $x(t)$ and how closely it matches the ideal uniform distribution.

B.4.3.1 Distribution Function

The probability distributions based on the first 100,000 values of each generator were obtained by determining the extents of the actual intervals of *x(t)* and *y(t)*, dividing the intervals into 1000 equally-sized intervals (bins), and then counting the frequency of x (or y) in those bins.

B.4.3.2 Mean and Variance

The mean μ and variance σ^2 were computed for *x(t)* and *y(t)* normalized to a unit interval, and appear in Tables B-2 and B-3.

B.4.3.3 Uniqueness of Values and Sequence

The output of each generator was checked for the existence of duplicate points: if all values are unique, the sequence does not repeat within the first 100,000 points. Since uniqueness is not guaranteed for projections of fractals, some *values* may be repeated within *x(t)* (or *y(t)*) even though there are no repeated *sequences*, therefore, if any duplicates are found, their adjacent values are compared as well to determine if multiple sequences exist with the given interval. The tests for uniqueness involve $\sim 5 \times 10^9$ compares and require about 20 minutes for each generator on a 150 MHz Pentium computer.

For the Hénon attractor, some points lie so close to each other that they are indistinguishable when using 32-bit **float**s for *x(t)* or *y(t)*, however, using 64-bit **double**s allows all members of *x(t)* and *y(t)* to be uniquely resolved.

B.4.3.4 Computation Times

The average computation time on 486DX-33 computer for each generator is included in Tables B-2 and B-3 for completeness. The computation time is not a critical parameter in this thesis, but may be for other applications.

B.4.4 Measures of the Increments of x(t) and y(t)

The measures of the increments $|x(t) - x(t-1)|$ of $x(t)$ are measures of the randomness of the order or sequencing of values in $x(t)$. From this the persistence of $x(t)$ can be determined and compared to white, pink, brown and black noise. Why is this important if we know that $x(t)$ is characterized by a uniform distribution? Consider a "random" variable R which has the following sequence:

$$R(t) = 0,1,2,3,0,1,2,3,0,1,2,3,... \qquad \text{(B-16)}$$

Each of the 4 possible values (0,1,2,3) have a uniform probability $p_t = 0.25$ of appearing, but the output is definitely *not* random! Therefore, *the degree of randomness* of a generator is important in addition to its *probability distribution.*

B.4.4.1 Distribution of Increments

In a manner similar to that described in Sec. B.4.3.1, the distribution of the increments between any two random numbers $R(t)$ and $R(t-1)$ for $0 \leq t < 100{,}000$, was obtained for each generator considered. The distribution is indicative of the persistence of $R(t)$: if the distribution is skewed toward small increments then $R(t)$ has a high persistence similar to brown or black noise, if the distribution is skewed toward large increments then $R(t)$ has a low persistence, similar to pink or white noise [Kins94f].

Table B - 1 shows the relationships between the classifications of noise "colours", the Hurst exponent H for fractional Brownian motion, the power spectra exponent β, and the variance dimension D_σ.

Table B - 1: Relationships of signal measures.

noise classification	β	H	D_σ
white	0	-0.5	2.5
pink	1	0.0	2.0
brown	2	+0.5	1.5
black	3	+1.0	1.0
black	4	+1.5	0.5

B.4.4.2 Variance Dimension of Increments

The variance dimension D_σ (Sec. 2.5.9) can be employed to relate the randomness of the output $R(t)$ obtained from any generator to the classifications commonly associated with random signals in Tables B-2 and B-3. The fundamental property measured is the randomness of $R(t)$ and the variance dimension can be considered to be a figure of merit of the generator.

B.4.5 Other Measures

There are other measures, such as the chi-squared test, which can be used to measure the goodness of the randomness of pseudo-random generators, but these tests are beyond the scope of this thesis.

B.5. Evaluation

B.5.1 Complexity of Generator

The complexity of underlying process, or attractor, of the pseudo-random generators can only be measured by the application of Taken's method, which was beyond the scope of this thesis. As for the chaotic generators, if we ignore the 2-projections of the Lorenz and Rössler attractors, then there is a strong correlation between the topological dimension D_{HM} of the attractor in the x-y plane, and the measure of the randomness of the sequence of $x(t)$ by means of the variance dimension D_σ . The Gingerbread Man and the Ikeda attractors ranked the highest topological dimensions for the attractors embedded in 2 dimensions. The 2-dimensional projection of the Lorenz attractor had the overall highest topological dimension, however, it miserably fails other tests (Sec. B.5.4).

B.5.2 Distribution and Statistical Measures

All the pseudo-random generators have relatively uniform distributions of $x(t)$, and their normalized values of the mean and variance are within 0.292% and 0.475% respectively of the ideal values. As for the chaotic generators, the normalized means are within 20.7% of the ideal value of 0.5000 (excluding the Lorenz and Rössler attractors), and the variances are not even close to the ideal. The variances are very small, reflecting the pseudo-bell-shaped likeness of the $x(t)$ and $y(t)$ distributions. Perhaps this should not be surprising as the attractors are not symmetrical horizontally or vertically, resulting in clusters or varying densities of points for projections onto either axis. The Julia attractor,

although it appears to the eye to have a uniform horizontal density of points, is actually characterized by huge spikes in the probability distribution function, some attaining heights 90 times higher than average. This property could be used to account for random defects in the dielectric modelled by the dielectric breakdown model.

B.5.3 Uniqueness Measures

If we use **float**s then all the pseudo-random generators except for *Rand2 (81)* have some duplicate values. The standard *rand()* function has *at least* 1 duplicate sequence in the first 100,000 points which is not surprising as it only has 32,768 unique values. None of the chaotic generators had any duplicate values within the first 100,000 values.

B.5.4 Variance Dimension

Based on the results of the variance dimension which measures the randomness of a sequence, *Rand2*, *Rand2(81)*, and *Rand3* are the best pseudo-random generators, but just by a very small margin. Their slightly higher dimension is due to the additional complexity introduced by the use of shuffling or multiple generators.

As for the chaotic generators, the *x(t)* output of the Lorenz and Rössler generators ranks with black noise, and these generators are therefore to be avoided in our application; they just are not sufficiently random. The randomness of the *x(t)* output of the Julia, Hénon, Ikeda, and Lozi attractors correlates closely with the randomness of pink noise. The randomness of the *x(t)* output of the Gingerbread Man attractor approaches that of white noise, meaning that it generates the most random sequence of all the strange attractors considered. These results lead to one important realization: that the proper use

of strange attractors allows the controlled generation of specific "colours" of noise. Pseudo-random generators generally produce only pink noise.

B.6. Results

The following pages contain plots of the strange attractors evaluated, and plots of the distributions and variance fractal dimensions of the pseudo-random and chaotic generators evaluated.

Table B - 2: Characteristics of selected random generators.

	x(t)					
generator	mean μ_n	variance σ_n^2	duplicate values	duplicate sequences	var. dim. D_σ	execution time (μs)
rand	0.49947	0.082957	68756	Yes	2.00636	3
Rand0	0.50028	0.083196	0	No	2.00407	7
Rand1	0.50028	0.083187	0	No	2.00164	12
Rand2	0.50084	0.083018	212	No	2.00656	17
Rand2(81)	0.50068	0.083321	0	No	2.01050	12
Rand3	0.50072	0.083392	196	No	2.00914	3
Randq1	0.50146	0.082963	n/a	No	2.00364	3
Randq2	0.49909	0.082938	n/a	No	2.00357	3
Gaussian	-.001475	0.996267	59	No	2.00896	~84

Table B - 3: Characteristics of selected chaotic generators.

generator	top. dim.	x(t)				y(t)					execution
	D_{HM}	mean μ_n	variance σ_n^2	duplicate values	duplicate sequences	var. dim. D_σ	mean μ_n	variance σ_n^2	duplicate values	duplicate sequences	time (µs)
Hénon	1.290	0.60310	0.079321	0	0	2.04055	0.60311	0.079320	0	0	6
Lozi	1.427	0.55942	0.053776	0	0	2.12349	0.55942	0.053775	0	0	7
Gingerbread Man	1.759	0.43479	0.054992	63358	0	2.34334	0.43479	0.054992	63358	0	6
Julia	1.223	0.50078	0.107336	0	0	2.00630	0.49981	0.065965	0	0	21
Ikeda	1.709	0.50027	0.049811	0	0	2.09430	0.57737	0.060152	0	0	56
Lorenz	1.797	0.50403	0.045712	n/a	n/a	0.99516	0.50976	0.030037	n/a	n/a	56
Rössler	1.550	0.44799	0.049829	n/a	n/a	0.69946	0.54136	0.055065	n/a	n/a	50

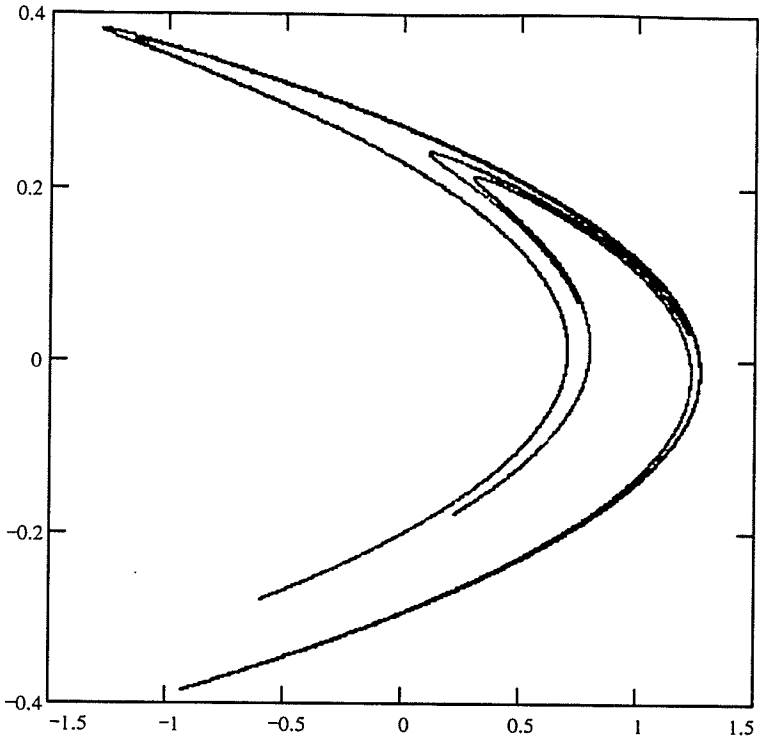


Figure B-1: Henon attractor

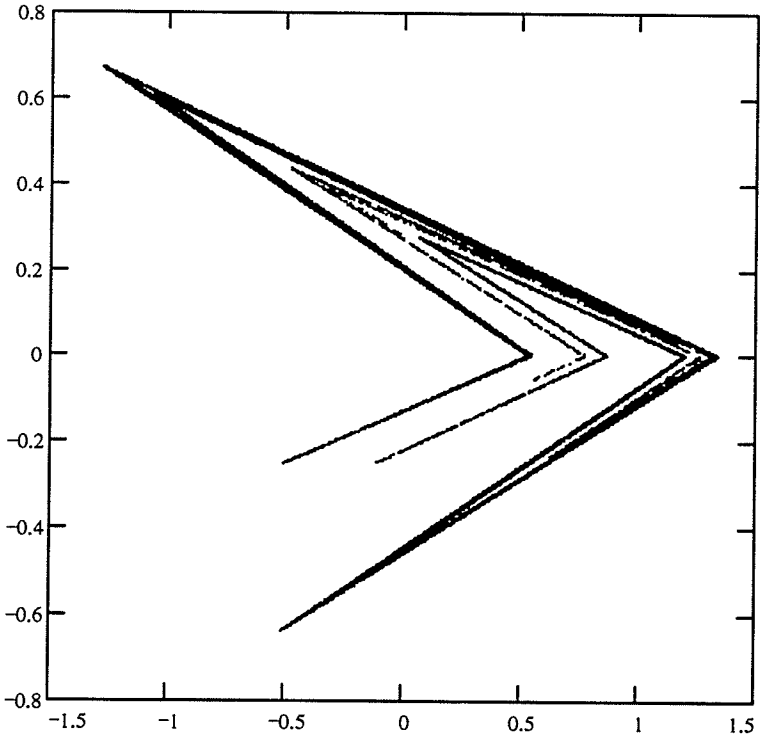


Figure B-2: Lozi attractor

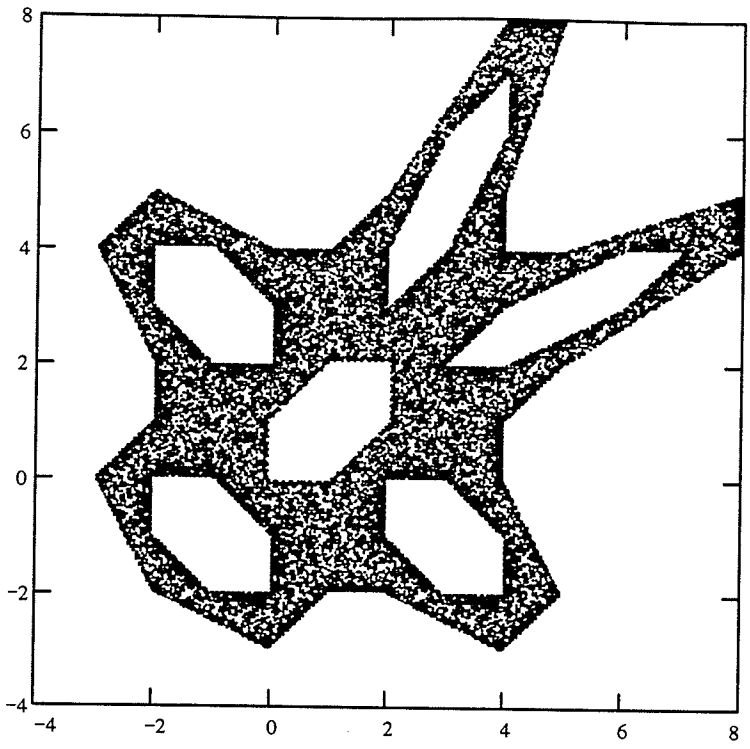


Figure B-3: Gingerbread Man attractor

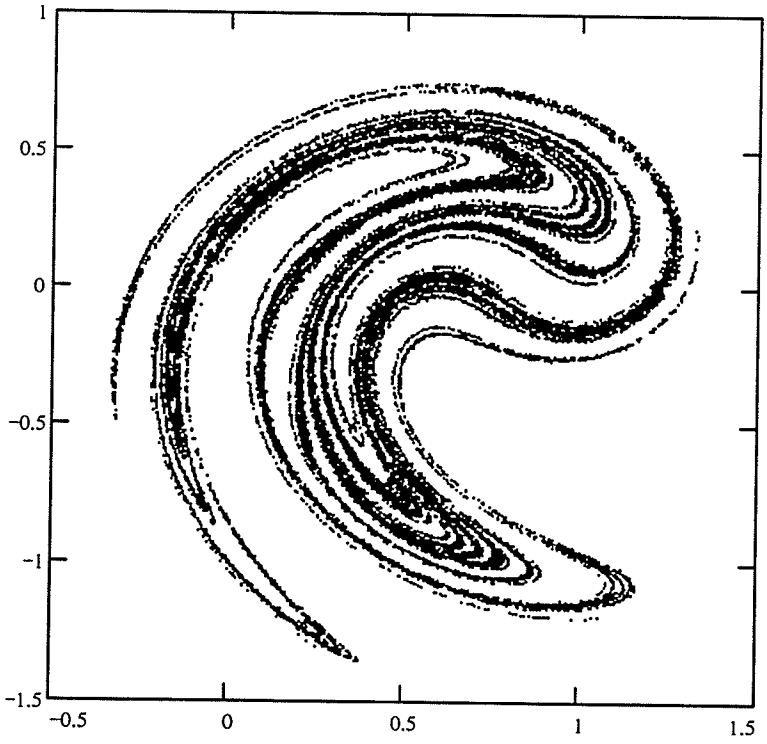


Figure B-4: Ikeda attractor

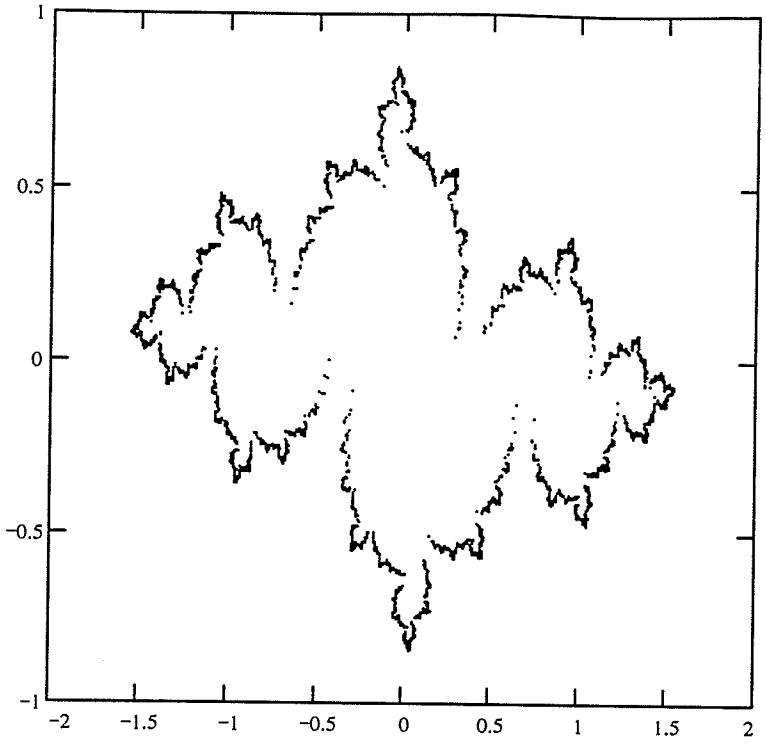


Figure B-5: Julia attractor

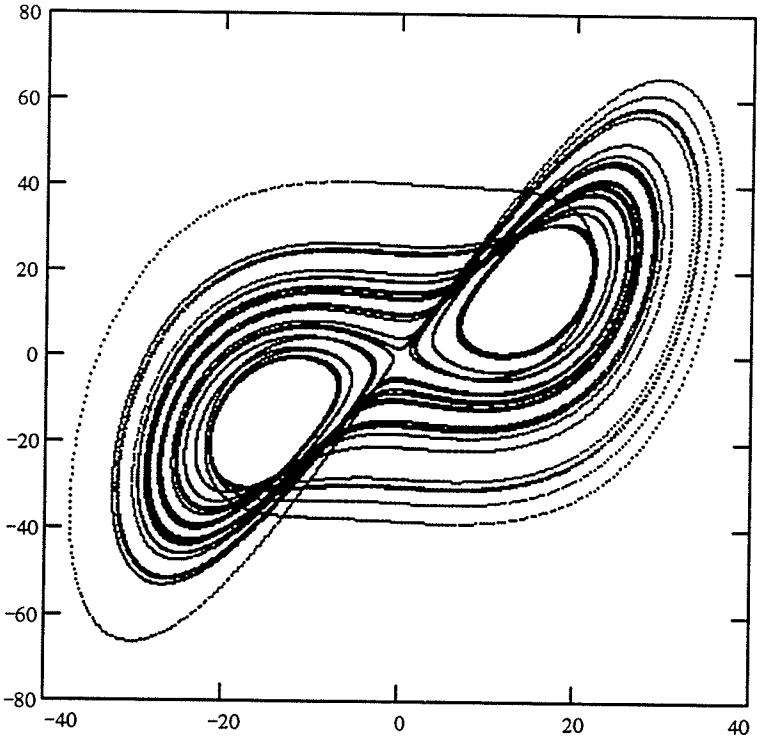


Figure B-6: Lorenz attractor (2-D)

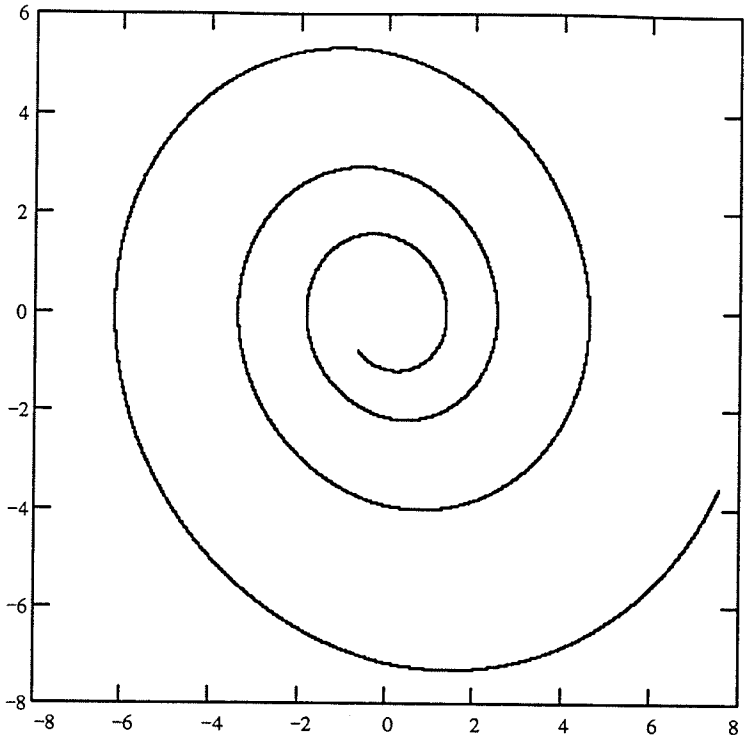


Figure B-7: Rossler attractor (2-d)

[This page intentionally left blank.]

Measures of Rand()

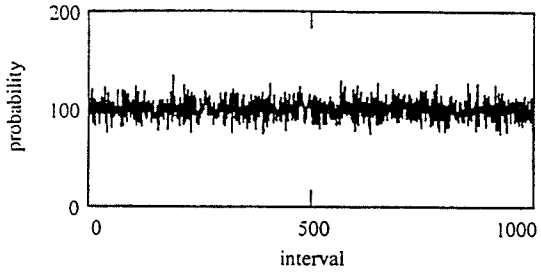


Figure B-8: Distribution of x(t)

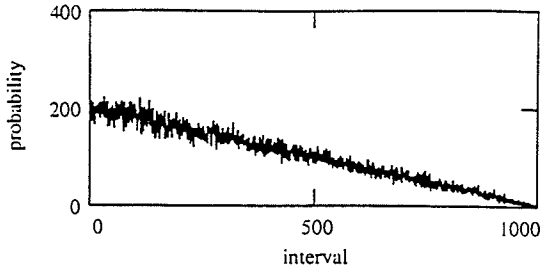


Figure B-9: Distribution of x(t)-x(t-1)

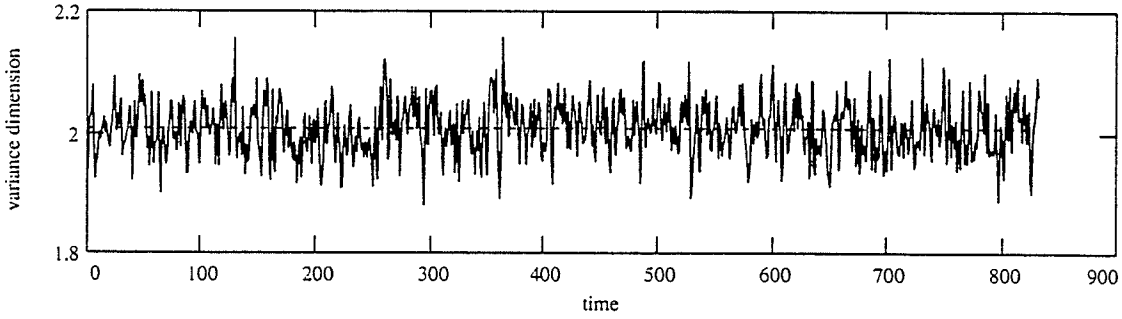


Figure B-10: Variance dimension

Measures of Rand0()

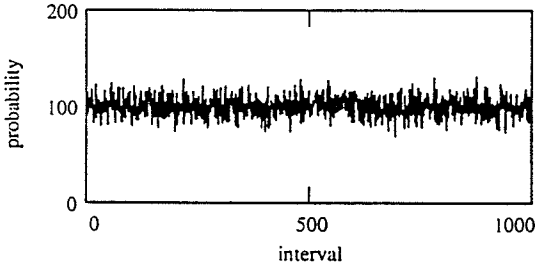


Figure B-11: Distribution of x(t)

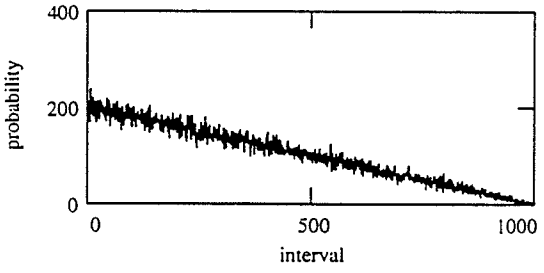


Figure B-12: Distribution of x(t)-x(t-1)

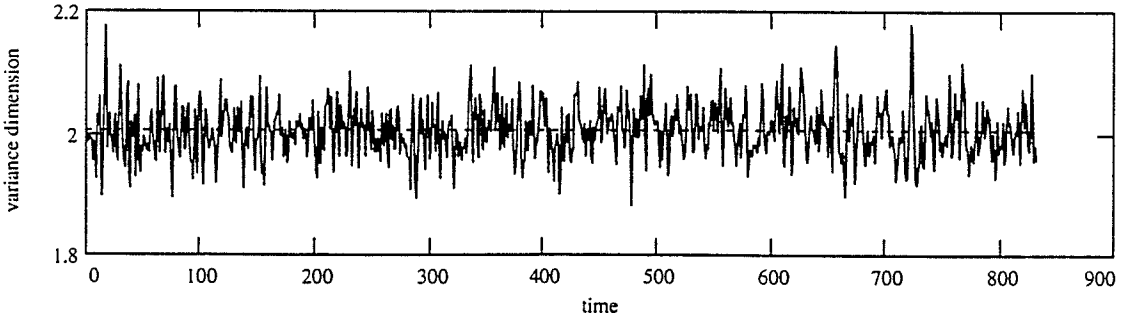


Figure B-13: Variance dimension

Measures of Rand1()

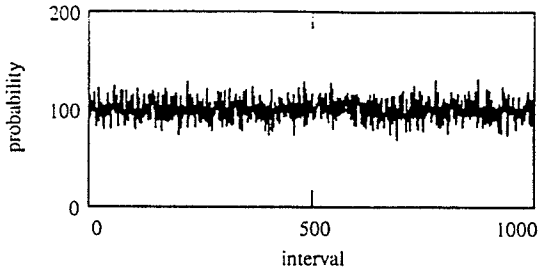


Figure B-14: Distribution of x(t)

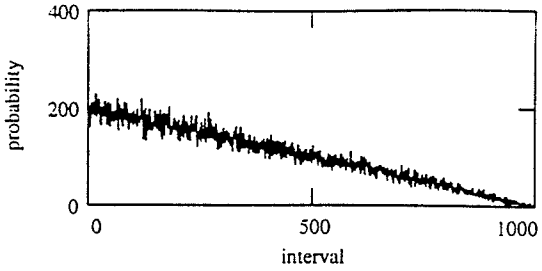


Figure B-15: Distribution of x(t)-x(t-1)

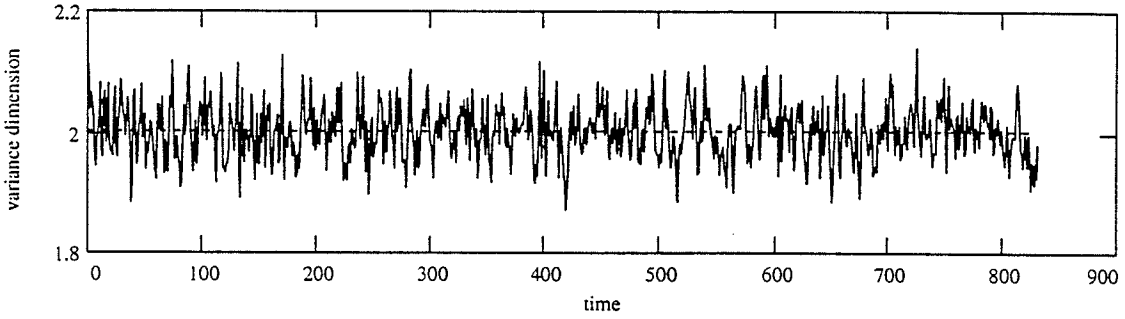


Figure B-16: Variance dimension

Measures of Rand2()

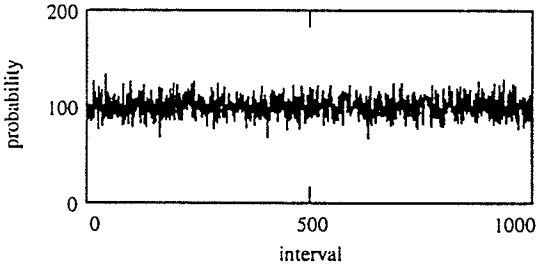


Figure B-17: Distribution of x(t)

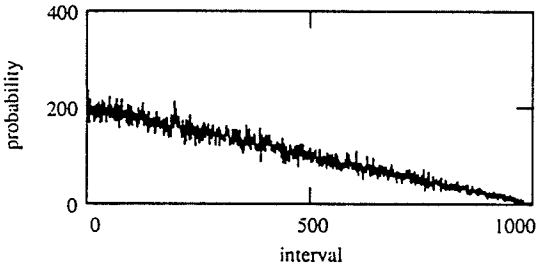


Figure B-18: Distribution of x(t)-x(t-1)

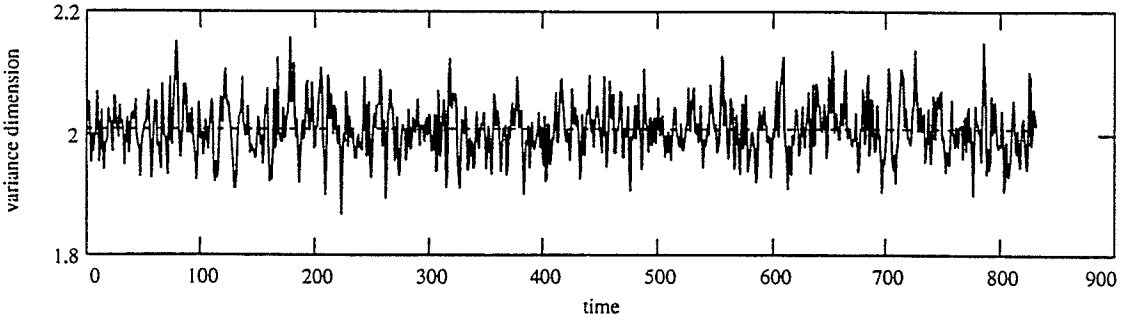


Figure B-19: Variance dimension

Measures of Rand2.81()

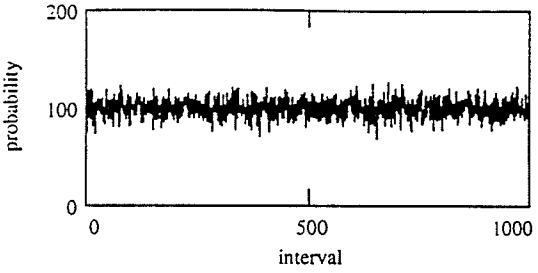


Figure B-20: Distribution of x(t)

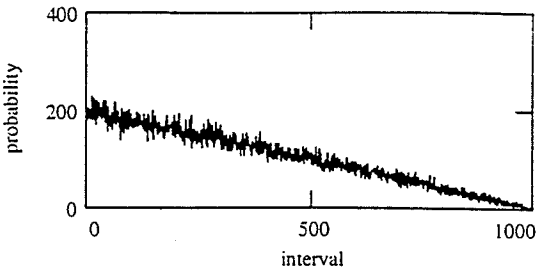


Figure B-21: Distribution of x(t)-x(t-1)

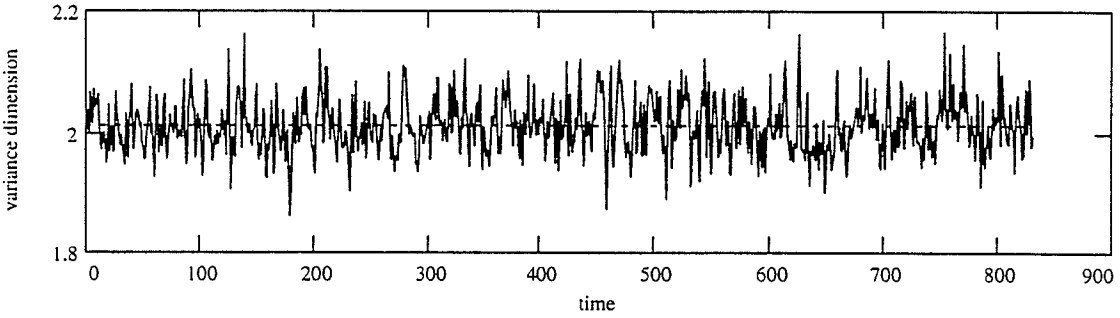


Figure B-22: Variance dimension

Measures of Rand3()

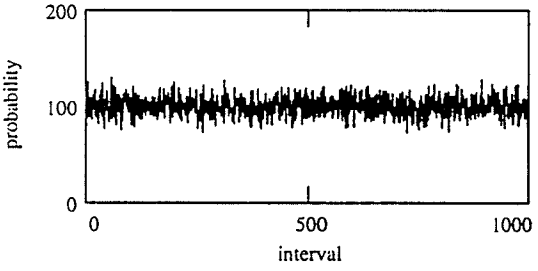


Figure B-23: Distribution of x(t)

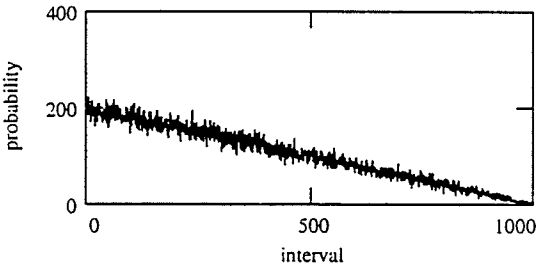


Figure B-24: Distribution of x(t)-x(t-1)

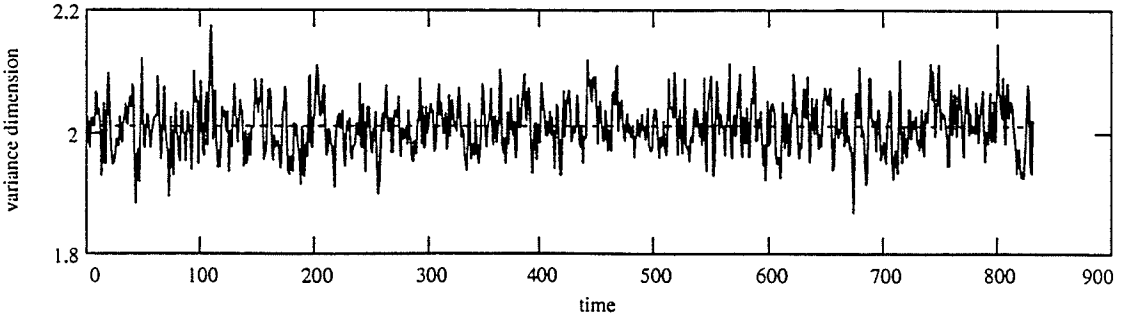


Figure B-25: Variance dimension

Measures of Randq1()

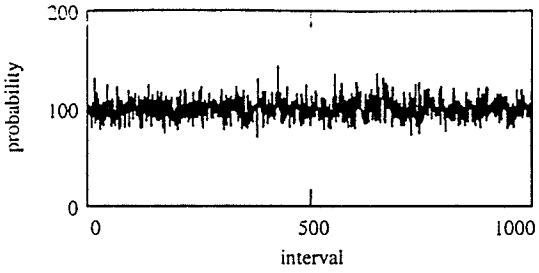


Figure B-26: Distribution of x(t)

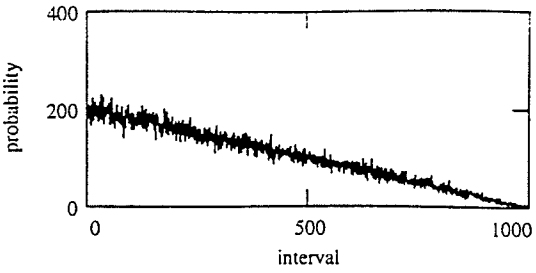


Figure B-27: Distribution of x(t)-x(t-1)

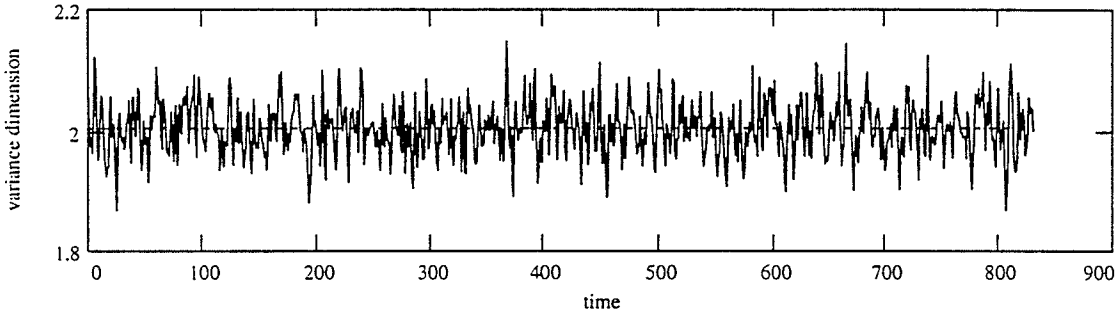


Figure B-28: Variance dimension

Measures of Randq2()

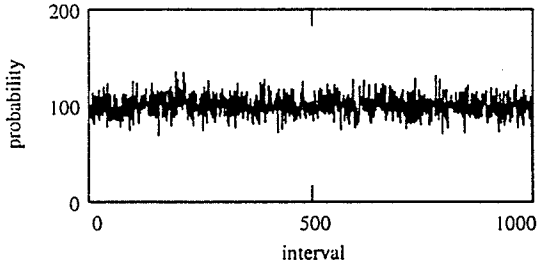


Figure B-29: Distribution of x(t)

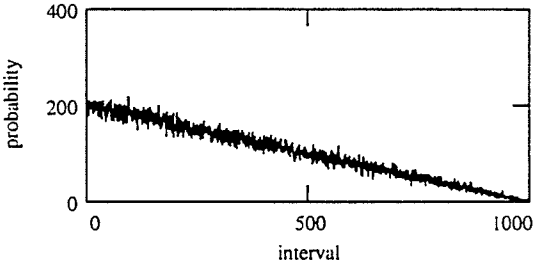


Figure B-30: Distribution of x(t)-x(t-1)

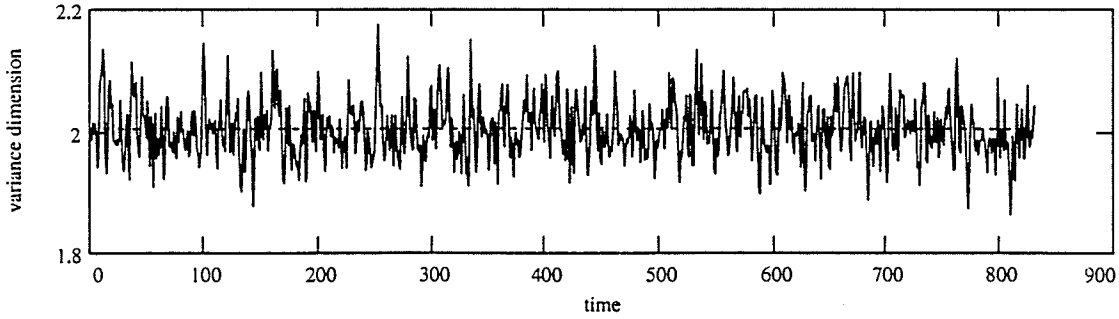


Figure B-31: Variance dimension

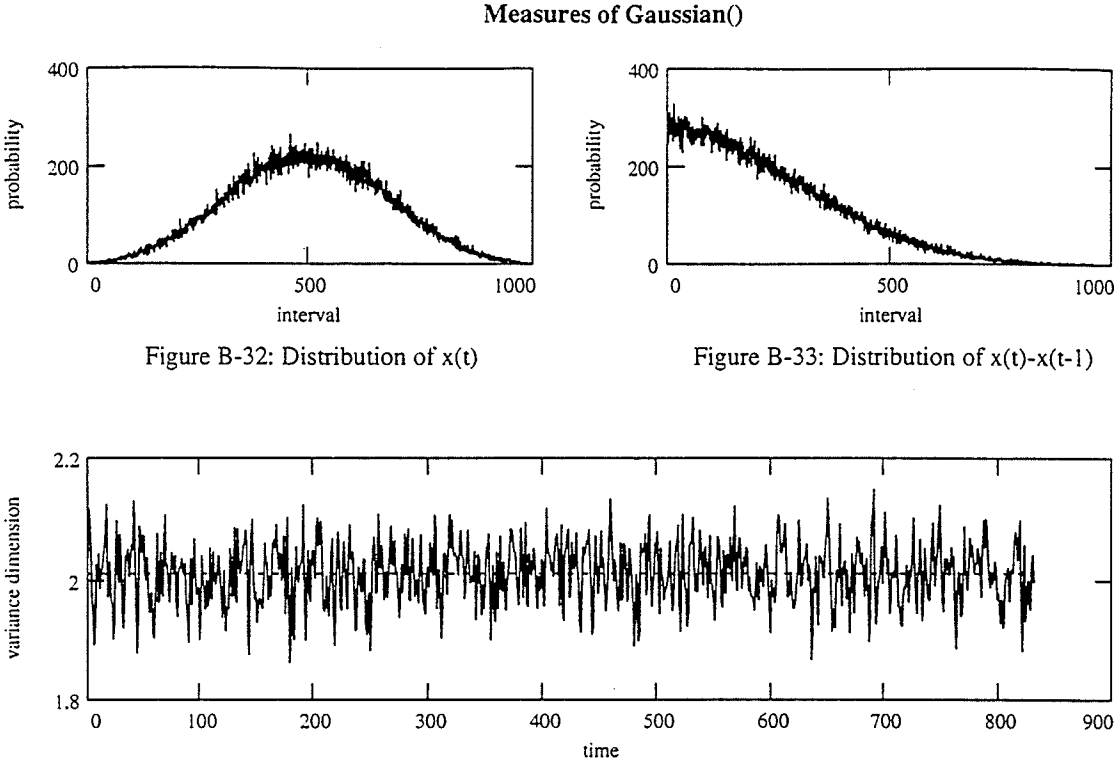


Figure B-32: Distribution of x(t)

Figure B-33: Distribution of x(t)-x(t-1)

Figure B-34: Variance dimension

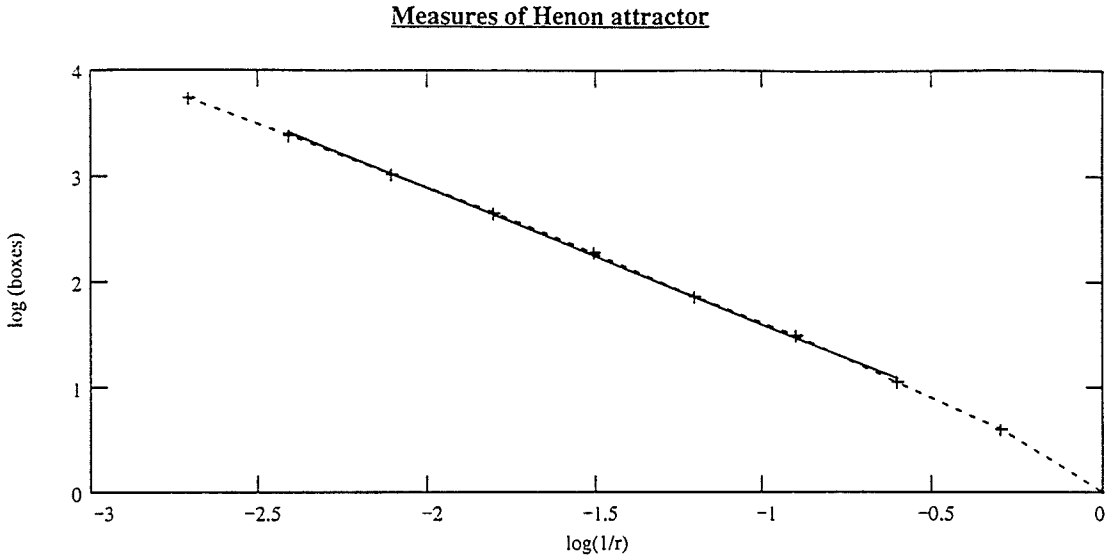


Figure B-35: Hausdorff dimension

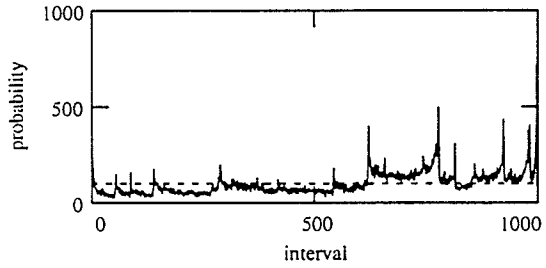


Figure B-36: Distribution of x(t)

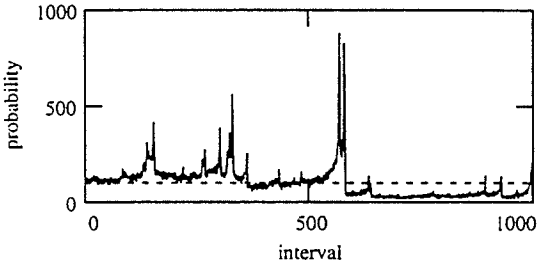


Figure B-37: Distribution of x(t)-x(t-1)

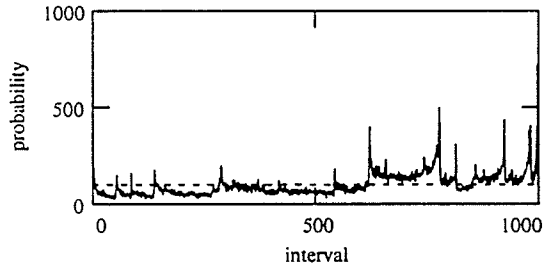


Figure B-38: Distribution of y(t)

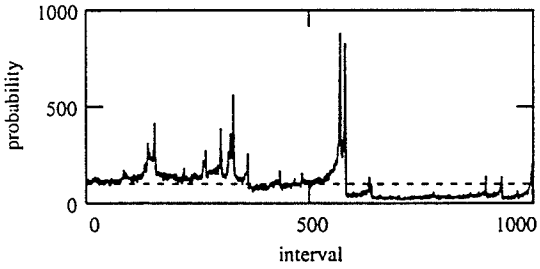


Figure B-39: Distribution of y(t)-y(t-1)

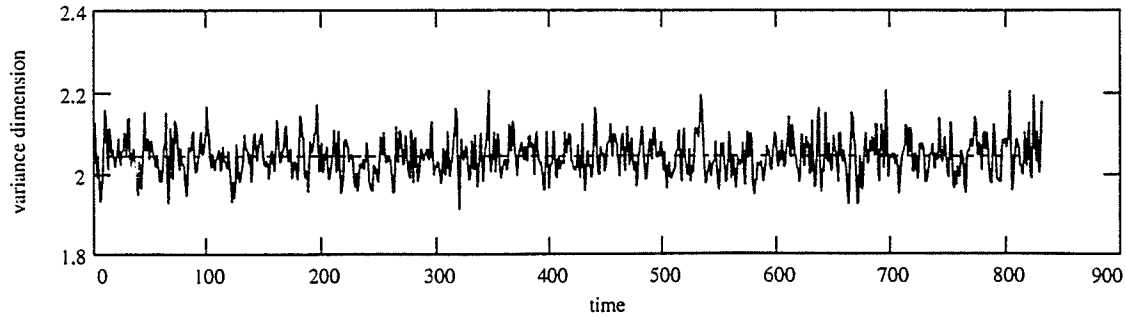


Figure B-40: Variance dimension

Measures of Lozi attractor

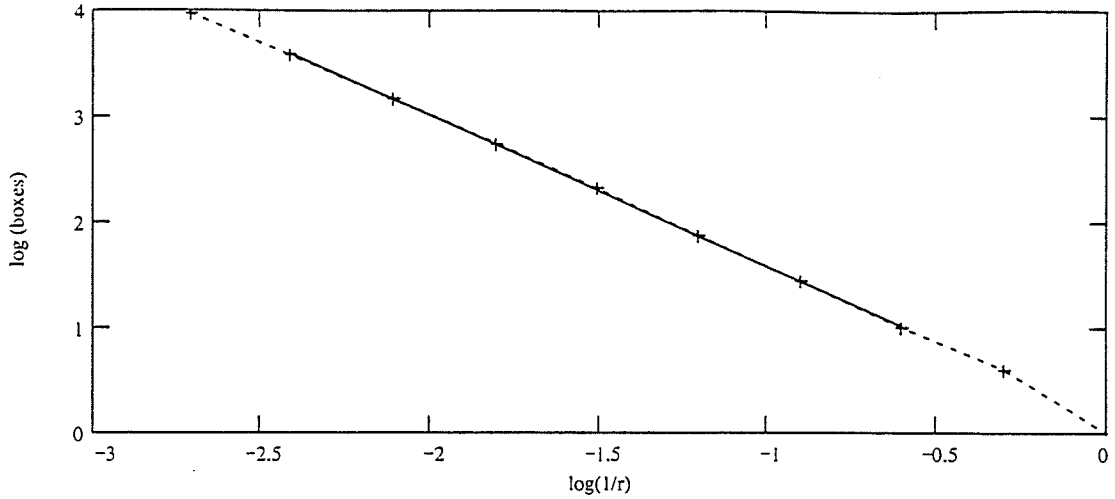


Figure B-41: Hausdorff dimension

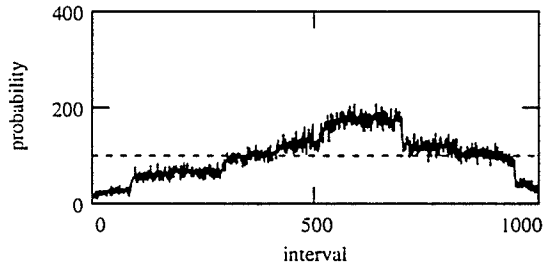


Figure B-42: Distribution of x(t)

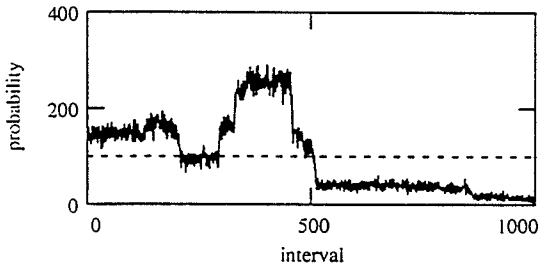


Figure B-43: Distribution of x(t)-x(t-1)

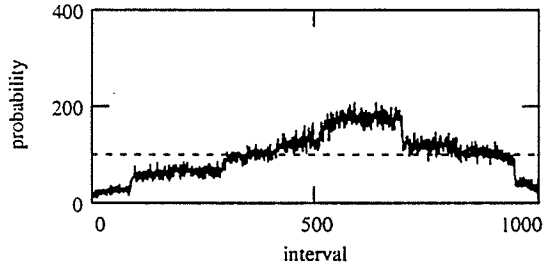


Figure B-44: Distribution of y(t)

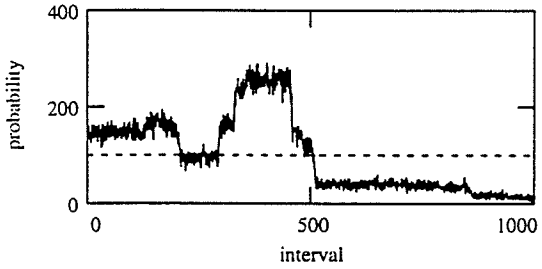


Figure B-45: Distribution of y(t)-y(t-1)

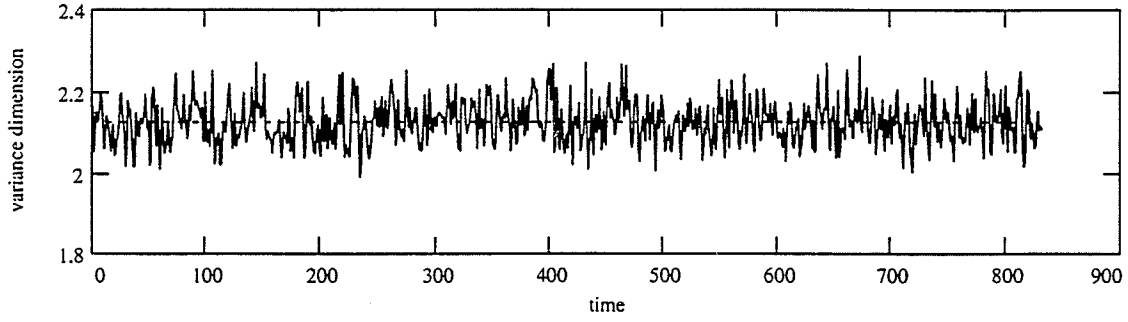


Figure B-46: Variance dimension

Measures of Gingerbread Man attractor

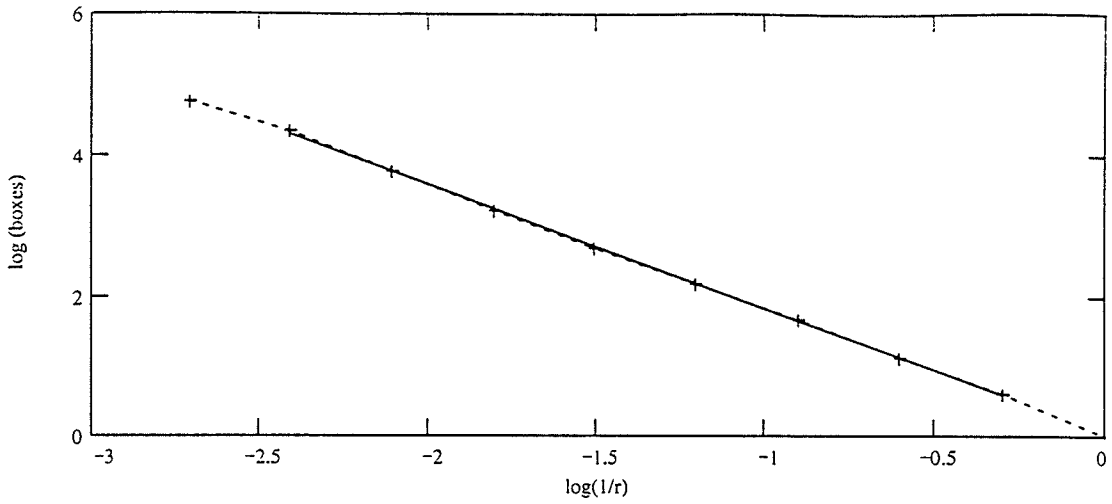


Figure B-47: Hausdorff dimension

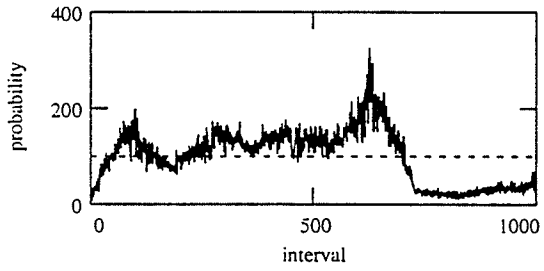


Figure B-48: Distribution of x(t)

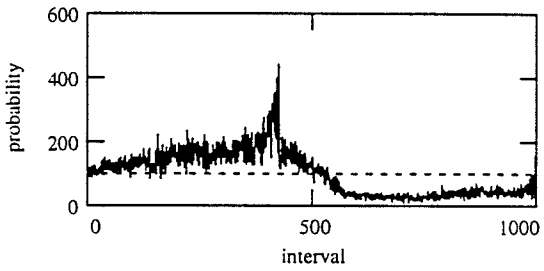


Figure B-49: Distribution of x(t)-x(t-1)

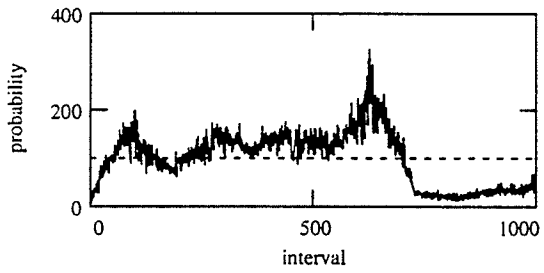


Figure B-50: Distribution of y(t)

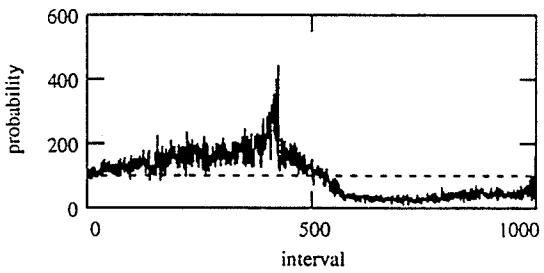


Figure B-51: Distribution of y(t)-y(t-1)

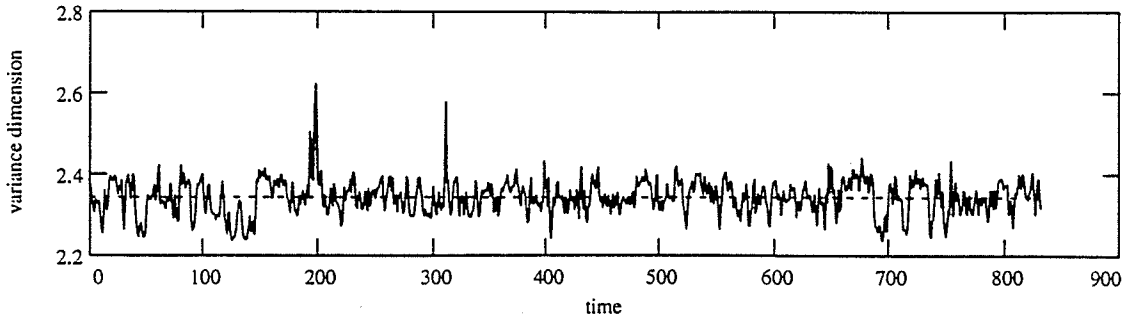


Figure B-52: Variance dimension

Measures of Ikeda attractor

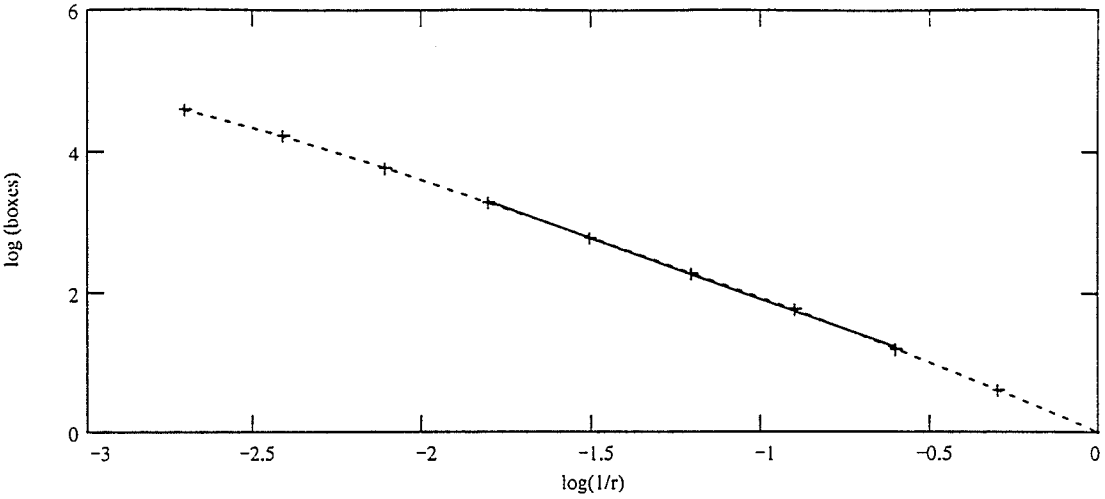


Figure B-53: Hausdorff dimension

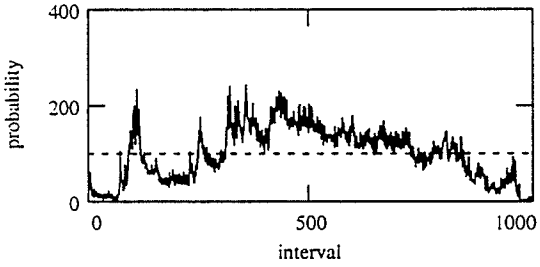


Figure B-54: Distribution of x(t)

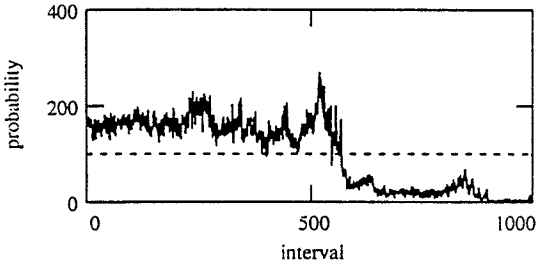


Figure B-55: Distribution of x(t)-x(t-1)

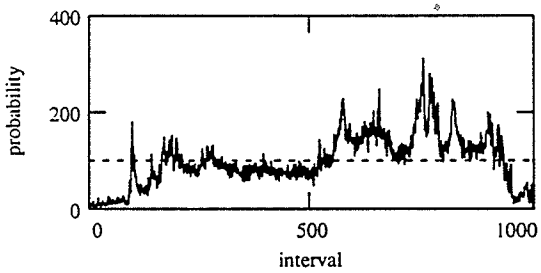


Figure B-56: Distribution of y(t)

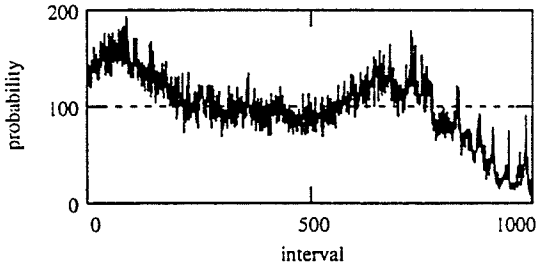


Figure B-57: Distribution of y(t)-y(t-1)

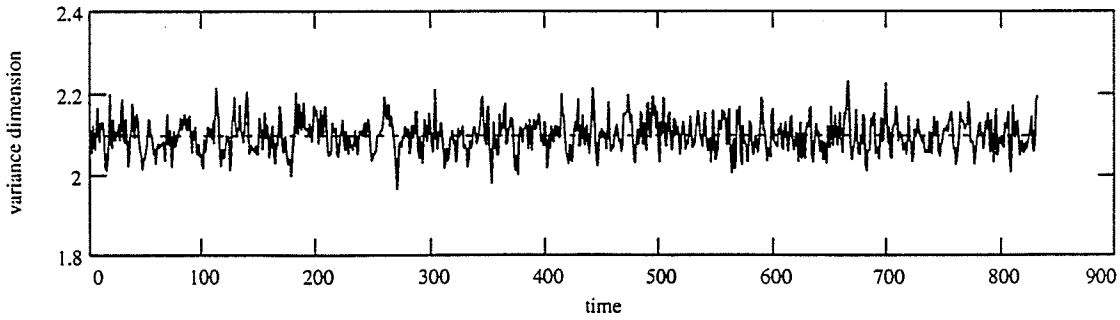


Figure B-58: Variance dimension

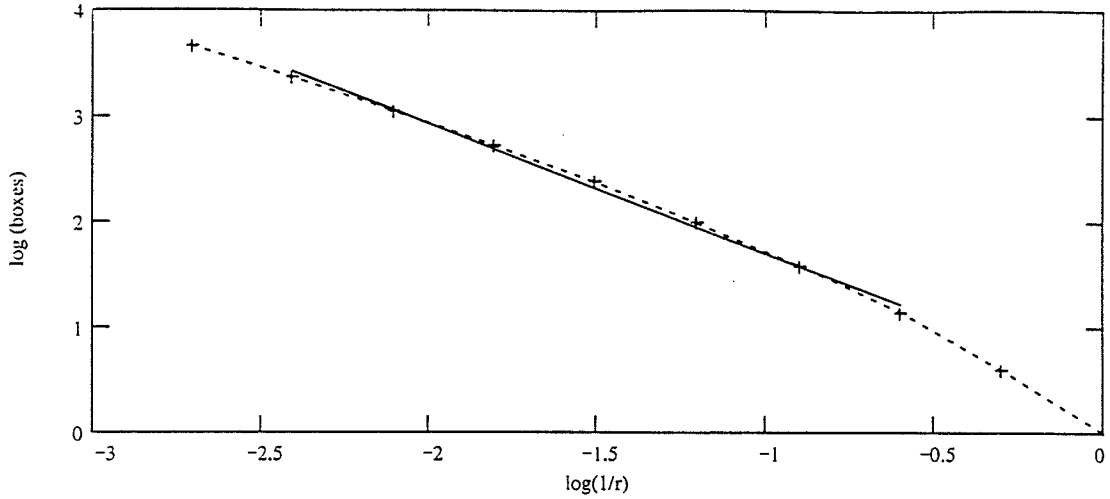


Figure B-59: Hausdorff dimension

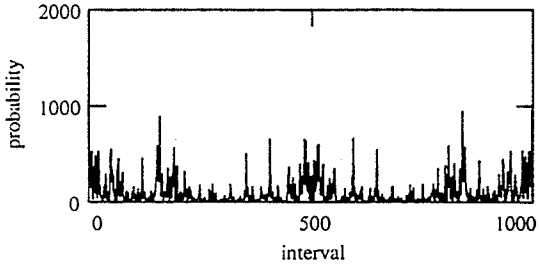


Figure B-60: Distribution of x(t)

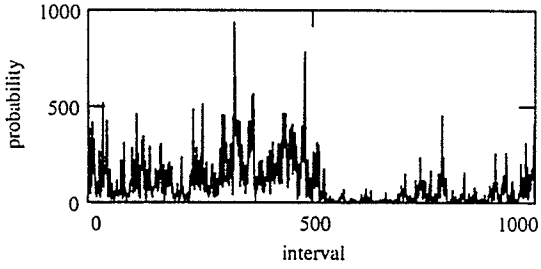


Figure B-61: Distribution of x(t)-x(t-1)

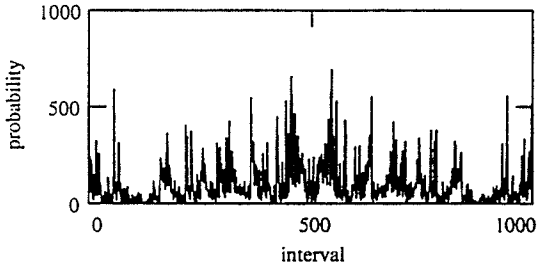


Figure B-62: Distribution of y(t)

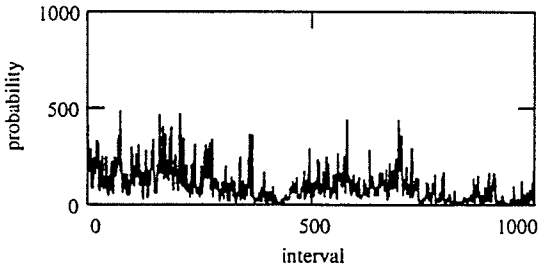


Figure B-63: Distribution of y(t)-y(t-1)

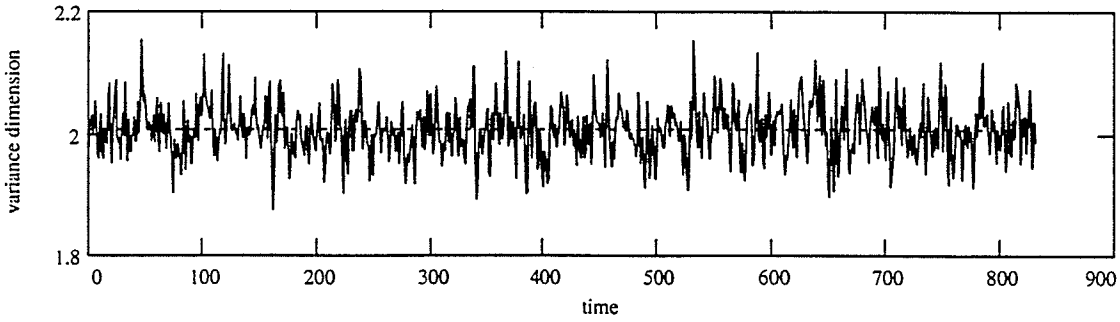


Figure B-64: Variance dimension

Measures of Lorenz attractor

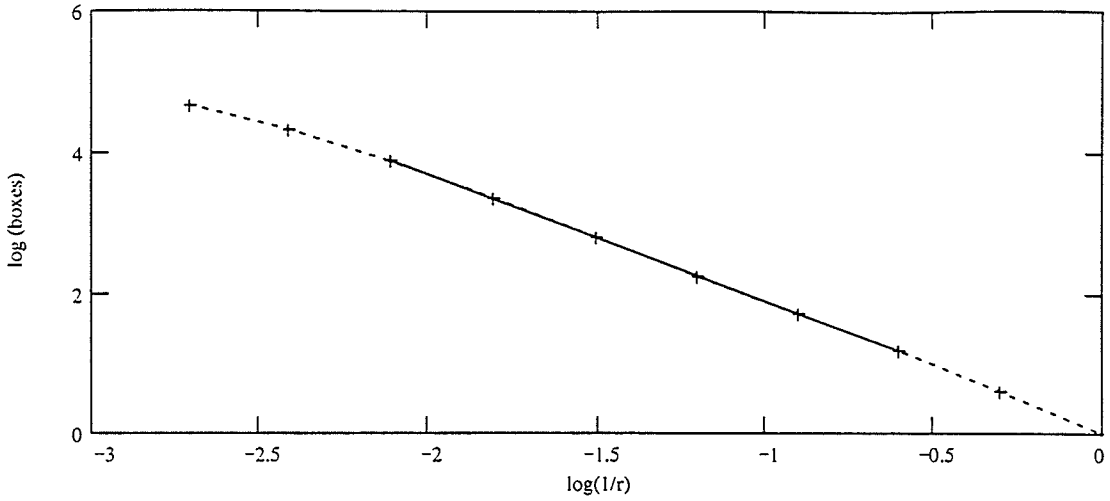


Figure B-65: Hausdorff dimension

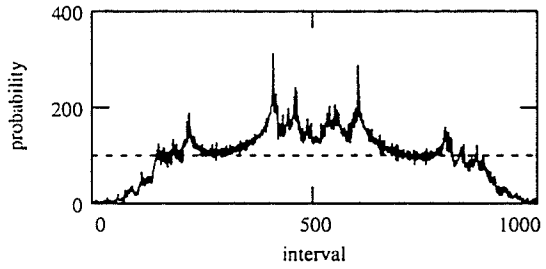


Figure B-66: Distribution of x(t)

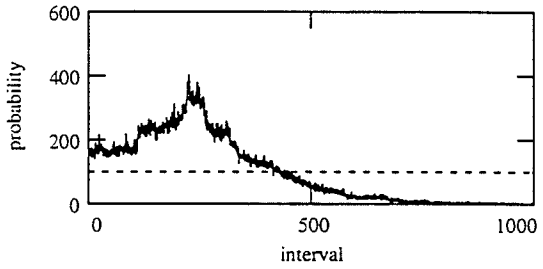


Figure B-67: Distribution of x(t)-x(t-1)

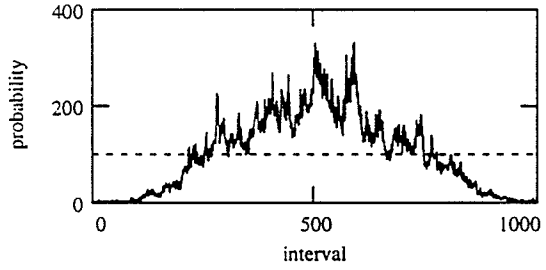


Figure B-68: Distribution of y(t)

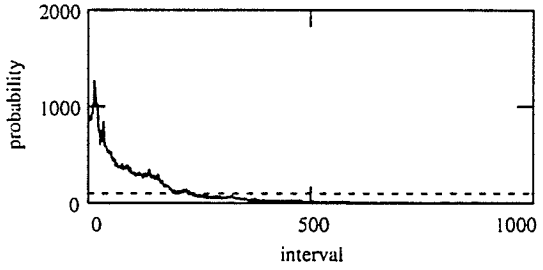


Figure B-69: Distribution of y(t)-y(t-1)

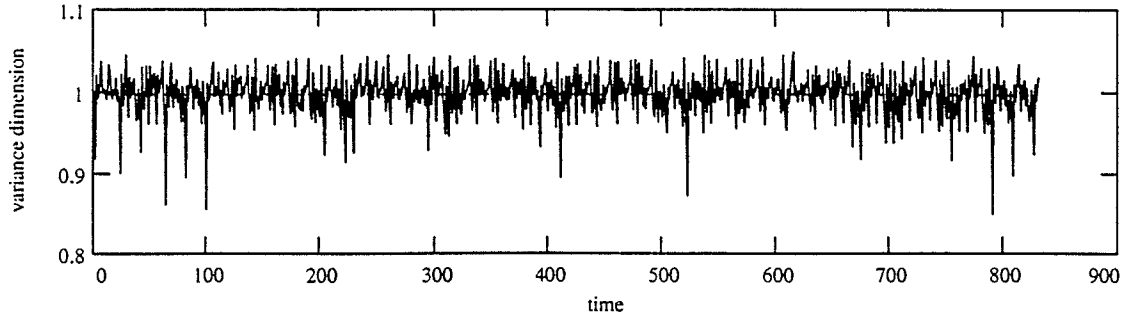


Figure B-70: Variance dimension

Measures of Rossler attractor

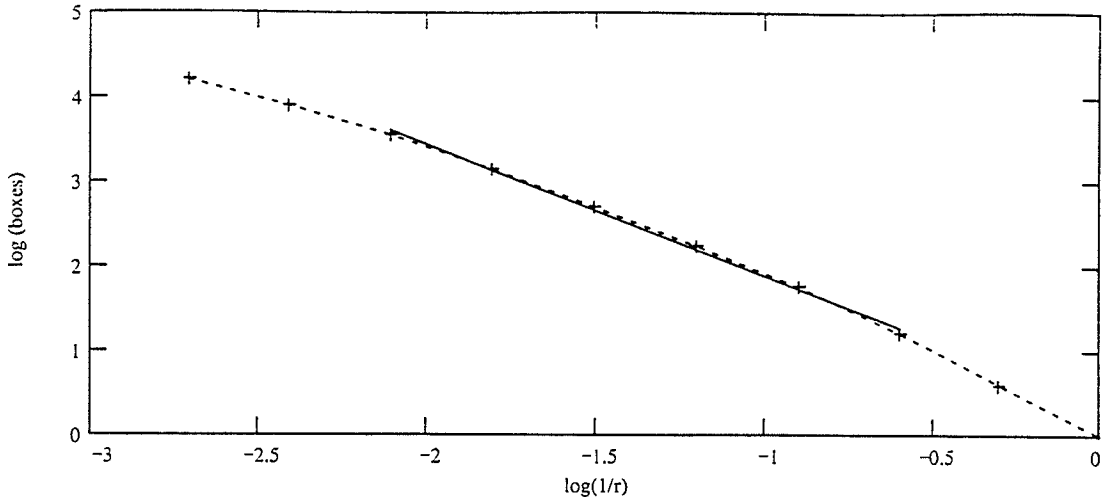


Figure B-71: Hausdorff dimension

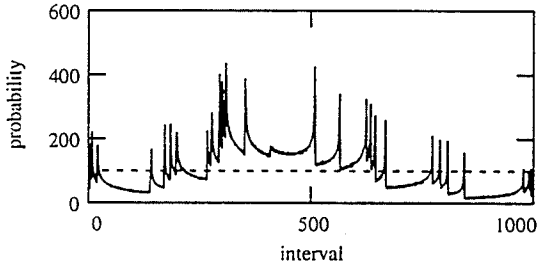


Figure B-72: Distribution of x(t)

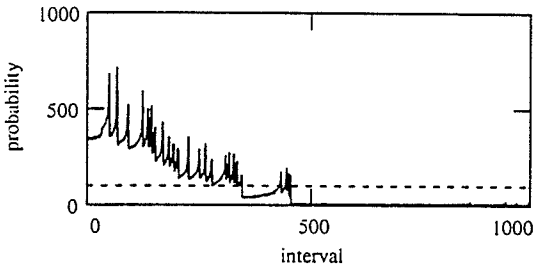


Figure B-73: Distribution of x(t)-x(t-1)

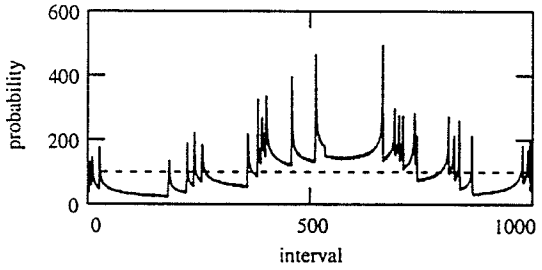


Figure B-74: Distribution of y(t)

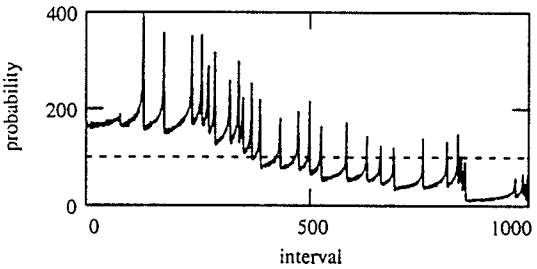


Figure B-75: Distribution of y(t)-y(t-1)

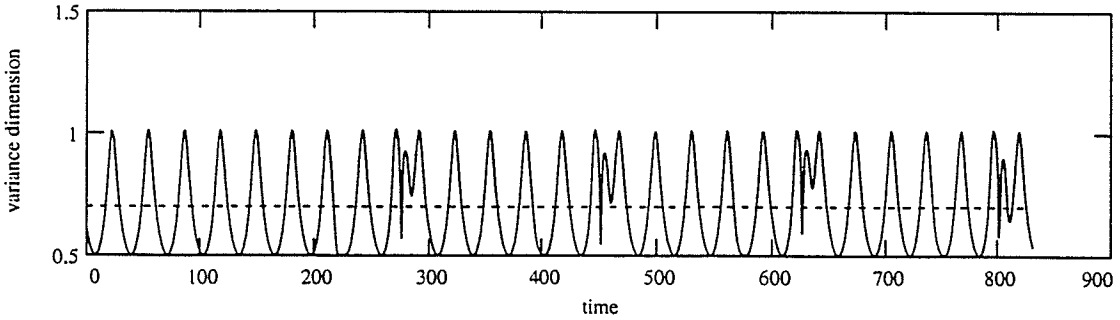


Figure B-76: Variance dimension

APPENDIX C

STENCILS FOR 2×2 BLOCK METHODS

C.1. Introduction

This appendix details the implementation of $n \times n$ block methods for the iterative solution of Laplace's equation on a discrete mesh, emphasizing the methodology for 2×2 blocks which can be extended to larger blocks. Blocks larger than 2×2 are not practical nor recommended for the solution of systems which allow "randomly-located" singularities in the interior of the mesh for reasons discussed in Sec. C-3.

C.1.1 Typical n×n Block of Points

If we consider the section of 5×6 points shown in Fig. C-1, which are part of a much larger mesh, then the application of block iterative methods allows groups or blocks of points to be simultaneously modified, whereas the application of point methods sequentially modifies individual points. Blocks are generally updated row-wise as in Successive Block Over-Relaxation (SBOR). Row-wise updates may be followed by column-wise updates as in Alternating Direction Successive Block Over-Relaxation (ADSBOR). Use of an acceleration factor increases the convergence rate of the method. The blocks can be overlapping or non-overlapping.

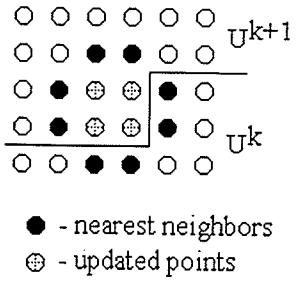


Fig. C - 1: Typical 2×2 block operation on a mesh.

C.1.2 Limitations on Boundaries

For the purposes of this thesis, the scope of the research has been restricted to the modelling of Laplacian fractals within the following configurations:

- parallel plates,
- coaxial with point or disc centre electrode,
- rectangular with disc centre electrode.

This leads to the issues of boundaries [Sec. 2.4.2]. During the solution for a given n×n block, all the *adjacent* points to it are considered to be momentarily fixed, thus being treated as *temporary* Dirichlet boundaries. In any configuration, all non-Dirichlet boundary points must be iterated. In the coaxial and rectangular configurations, the dielectric is completely enclosed by Dirichlet boundaries, therefore, the $n \times n$ blocks are restricted to the interior points, although some of the adjacent points may lie on the Dirichlet boundaries. For the parallel plates configuration, Neumann boundaries are defined along both vertical sides, and it should be clear that all $n \times n$ blocks along the either vertical side will be missing n adjacent points, forcing the outermost column of n points to have Neumann boundaries imposed on them. Additionally, for the configurations considered, blocks can not experience Neumann boundaries *along more*

than one side at a time. Figure C-2.a shows parallel plates with the solution of three 2×2 blocks: the leftmost block experiences Neumann boundaries on its leftmost 2 points, the other 2×2 blocks are solved with temporary or permanent Dirichlet boundaries assumed on all their adjacent points. Figure C-2.b shows the solution of two 2×2 blocks, both of which have temporary or permanent Dirichlet boundaries assumed on all their adjacent points.

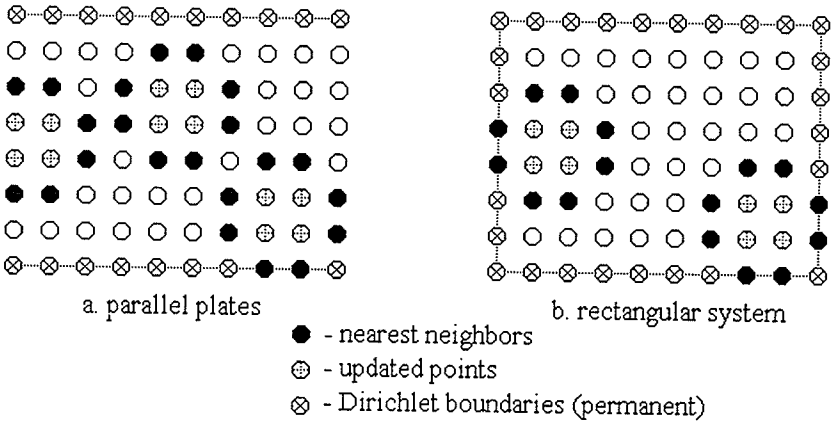


Fig. C - 2: Examples of Neumann and Dirichlet Boundaries.

C.1.3 Conventions

Consider the 2×2 block solution for points with unknown potentials represented by D, E, H, and J in Fig. C-3, and having adjacent points with potentials of A, B, C, F, G,K, L, and M at the k^{th} stage of iteration. It is desired to find the solutions D', E', H', and J' at iteration stage k+1. Figure C-3 will be the basis for all the equations for 2×2 block methods. The letters A through M are used for clarity and brevity instead of the longer $U_{i,j}^{k}$ notation, and the letters D', E', H', and J' are used instead of the $U_{i,j}^{k+1}$ notation.

$$\begin{array}{cccc} & A & B & \\ C & D & E & F \\ G & H & J & K \\ & L & M & \end{array}$$

Fig. C - 3: An example 2×2 block with potentials of adjacent points.

C.1.4 Typical Solution of 2×2 Blocks

C.1.4.1 Non-accelerated Solution with Dirichlet Boundaries

If we write the equations for the 5-point approximations of solutions D', E', H', and J' for the k+1 stage of iteration, we obtain

$$\begin{aligned} 4D' &= A + C + E' + H' \\ 4E' &= B + F + D' + J' \\ 4H' &= G + L + D' + J' \\ 4J' &= K + M + E' + H' \end{aligned} \qquad \text{(C-1)}$$

Rearranging, we can write

$$\begin{aligned} 4D' - E' - H' &= A + C \\ 4E' - D' - J' &= B + F \\ 4H' - D' - J' &= G + L \\ 4J' - E' - H' &= K + M \end{aligned} \qquad \text{(C-2)}$$

Employing the following simplifications,

$$\begin{aligned} d &= A + C \\ e &= B + F \\ h &= G + C \\ j &= K + M, \end{aligned} \qquad \text{(C-3)}$$

Eqs. C-2 can be also expressed in matrix form as

$$\begin{vmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{vmatrix} \begin{vmatrix} D' \\ E' \\ H' \\ J' \end{vmatrix} = \begin{vmatrix} d \\ e \\ h \\ j \end{vmatrix} \tag{C-4}$$

The implicit solutions of D', E', H', and J' in terms of points adjacent to the block can then be expressed as

$$\begin{aligned} D' &= \tfrac{1}{24}(7d + 2e + 2h + j) \\ E' &= \tfrac{1}{24}(2d + 7e + h + 2j) \\ H' &= \tfrac{1}{24}(2d + e + 7h + 2j) \\ J' &= \tfrac{1}{24}(d + 2e + 2h + 7j) \end{aligned} \tag{C-5}$$

Equations C-5 can be represented by the following, rotatable computational stencil for any point X shown in Fig. C-4.

$$\begin{array}{ccc} & +\frac{1}{12} \quad +\frac{1}{24} & \\ +\frac{1}{12} & & +\frac{1}{24} \\ +\frac{7}{24} & X & +\frac{1}{12} \\ & +\frac{7}{24} \quad +\frac{1}{12} & \end{array}$$

Fig. C - 4: Rotatable, non-accelerated, 2×2 stencil.

C.1.4.2 Accelerated Solution with Dirichlet Boundaries

If it is desired to accelerate the convergence of the iterative process, then the 5-point SOR equations for the 4 unknown potentials at stage k+1 are

$$
\begin{aligned}
D' &= D + \frac{\omega}{4}(A + C + E' + H' - 4D) \\
E' &= E + \frac{\omega}{4}(B + F + D' + J' - 4E) \\
H' &= H + \frac{\omega}{4}(G + L + D' + J' - 4H) \\
J' &= J + \frac{\omega}{4}(K + M + E' + H' - 4J)
\end{aligned}
\tag{C-6}
$$

Rearranging the terms, Eqs. C-6 can be expressed in matrix form as

$$
\begin{vmatrix} 4 & -\omega & -\omega & 0 \\ -\omega & 4 & 0 & -\omega \\ -\omega & 0 & 4 & -\omega \\ 0 & -\omega & -\omega & 4 \end{vmatrix} \begin{vmatrix} D' \\ E' \\ H' \\ J' \end{vmatrix} = \begin{vmatrix} \omega(A+C) + 4D(1-\omega) \\ \omega(B+F) + 4E(1-\omega) \\ \omega(G+L) + 4H(1-\omega) \\ \omega(K+M) + 4J(1-\omega) \end{vmatrix}
\tag{C-7}
$$

The implicit solutions of D', E', H', and J' in terms of points adjacent to the block can then be expressed as

$$
\begin{aligned}
D' &= \frac{1}{8(4-\omega^2)}\left[(8-\omega^2)d + 2\omega e + 2\omega h + \omega^2 j\right] \\
E' &= \frac{1}{8(4-\omega^2)}\left[2\omega d + (8-\omega^2)e + \omega^2 h + 2\omega j\right] \\
H' &= \frac{1}{8(4-\omega^2)}\left[2\omega d + \omega^2 e + (8-\omega^2)h + 2\omega j\right] \\
J' &= \frac{1}{8(4-\omega^2)}\left[\omega^2 d + 2\omega e + 2\omega h + (8-\omega^2)j\right]
\end{aligned}
\tag{C-8}
$$

where

$$
\begin{aligned}
d &= \omega(A + C) + 4D(1-\omega) \\
e &= \omega(B + F) + 4E(1-\omega) \\
h &= \omega(G + L) + 4H(1-\omega) \\
j &= \omega(K + M) + 4J(1-\omega)
\end{aligned}
\tag{C-9}
$$

The implementation of accelerated block methods results in more complicated expressions to evaluate, and hence require more effort than non-accelerated block

methods. Equations C-9 must be evaluated prior to Eqs. C-8 to eliminate repeated calculations of d, e, h, and j.

C.1.5 Complications Due to Enclosed Singularities

Interior mesh points which have fixed potential values are known as *singular points*, or *singularities*, and their fixed values as *singular values*. In our case, singular points are those having a potential of either 0 or the maximum electrode potential. If a 2×2 block under consideration encloses any number of singularities, that is points with known *potentials* for our application, then the equations must be modified accordingly. Referring to Fig. C-3, if D is a singular value, we cannot use just the remaining three equations for E', H' and J' from Eqs. C-5 or Eqs. C-8. If *any* singularities are enclosed *within* the block, then as *permanent* Dirichlet boundaries, they must be used instead of their adjacent points which normally are treated as *temporary* Dirichlet boundaries. This modifies the equations for the remaining non-singular points in the block.

This is perhaps best served by the example shown in Fig. C-5. Suppose that D is a singularity represented by •, then the remaining three non-singular potentials $\{E', H', J'\}$ in the block must be described not in terms of A and C, but in terms of D.

$$\begin{array}{cccc} & A & B & \\ C & \bullet & E & F \\ G & H & J & K \\ & L & M & \end{array}$$

Fig. C - 5: Example of 2×2 block enclosing a singularity at D.

Derivations of non-rotatable stencils for all possible permutations of enclosed singularities appear in Sec. C - 2.1.

C.1.6 Complications Due to Imposed Neumann Boundaries

If a 2×2 block under consideration lies along an edge so that Neumann boundaries are imposed upon any of the points within the block, similar to the leftmost block in Fig. C-2.a, then the 5-point operator equations must be modified accordingly (Sec. 2.4.2.3).

C.1.7 Solution Approaches

C.1.7.1 Generalized Inverse Matrix

The first attempt to develop a unified approach to automatically handle enclosed singularities, handle imposed Neumann boundaries, and implement an acceleration factor was to derive a generalized inverse matrix. In reality, an $n \times n$ block encloses n^2 points of which we assume m points to be singularities, which reduces the original $n^2 \times n^2$ resultant matrix an $(n^2\text{-}m) \times (n^2\text{-}m)$ matrix. The generalized matrix inverse is always of size $n^2 \times n^2$, so four Kronecker-delta type variables α, β, χ, and δ are used to enable or remove by multiplication of 1 or 0, the rows and columns in the *inverse* matrix corresponding to some singularity x and the contributions of x to the inverse terms for the other variables. If the x-related terms are zeroed in the original matrix prior to its inverse, then that matrix becomes singular and its inverse is undefined.

Consider a generalized inverse matrix M^{-1} of coefficients of Eqs. C-2 with the inclusion of the delta variables for the non-accelerated, Dirichlet-only case,

$$M^{-1} = \frac{\begin{bmatrix} \alpha(-16+\delta(\beta+\chi)) & -4\alpha\beta & -4\alpha\chi & -\alpha\delta(\beta+\chi) \\ -4\alpha\beta & \beta(-16+\chi(\alpha+\delta)) & -\beta\chi(\alpha+\delta) & -4\beta\delta \\ -4\alpha\chi & -\beta\chi(\alpha+\delta) & \chi(-16+\beta(\alpha+\delta)) & -4\chi\delta \\ -\alpha\delta(\beta+\chi) & -4\beta\delta & -4\chi\delta & \delta(-16+\alpha(\beta+\chi)) \end{bmatrix}}{Det(M)},$$

$$Det(M) = -16+(\alpha+\delta)(\beta+\chi)\cdot \qquad \text{(C-10)}$$

The inverse matrix has simple terms if Neumann boundaries and acceleration is ignored. There are several problems with this approach. First, the complexity of the generalized inverse matrix capable of handling Neumann boundaries on either side and any number of enclosed singularities plus acceleration is prohibitive from a computational and performance perspective. Secondly, the majority of blocks are "uncontaminated" by singularities, so the extra work such as the multiplication of the delta variables imposed by the generalization exacts a significant time penalty.

C.1.7.2 Implicit Methods

In our situation, we have 3 possible configurations of boundaries surrounding or imposed on the points of a 2×2 block:

- Neumann boundaries imposed on the leftmost points of the block, Dirichlet boundaries adjacent to the other 3 sides, (Fig. C-6.a),
- Dirichlet boundaries around all 4 sides of the block (Fig. C-6.b), and
- Neumann boundaries imposed on the rightmost points of the block, Dirichlet boundaries adjacent to the other 3 sides (Fig. C-6.c).

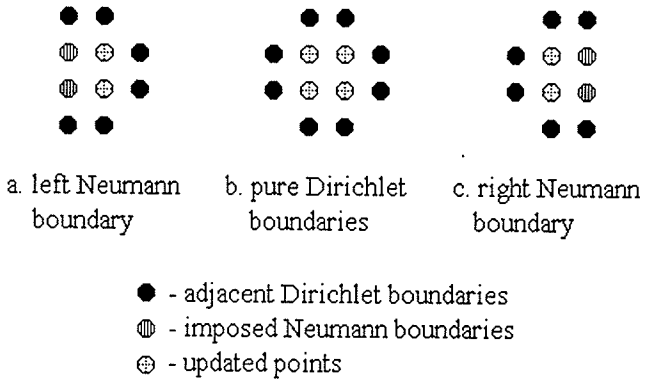


Fig. C - 6: Boundaries experienced by a 2×2 block.

Additionally, the 2×2 block can enclose any one of 2^4 combinations of singularities. Altogether, there are $3 \times 2^4 = 48$ combinations of possible scenarios.

C.1.7.2.1 Rotatable and Non-rotatable Stencils

To compute all possible combinations of singularities, a minimum of 6 rotatable stencils are required, for a total of 18 stencils required to account for the 3 boundary configurations.

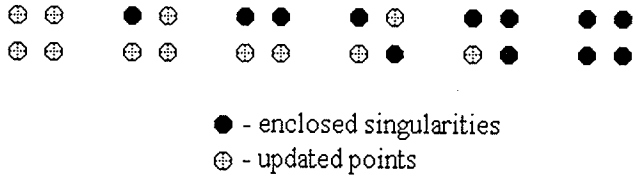
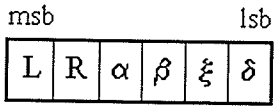


Fig. C - 7: Minimum required set of rotatable stencils.

Prior to using *any* stencil for a given block, a 6-bit *stencil index* is computed based on the number and position of enclosed singularities and based on the left and right boundary conditions of that block. This index is used to select the proper rotatable stencil.

The stencil index is computed as shown in Fig. C-8, where the appropriate bit is logic 1 if the corresponding condition is true.



L - left Neumann boundary
R - right Neumann boundary
α - D is a non-singular value
β - E is a non-singular value
χ - H is a non-singular value
δ - J is a non-singular value

Fig. C - 8: Composition of 6-bit stencil index.

It should be noted that extra effort is required by rotatable stencils to *implement the proper rotation* of the stencil. To eliminate this step from the solution process for each block, 48 unique, non-rotatable stencils are used, one for each possible scenario. This decreases the execution time at the acceptable expense of increased code.

The methodology presented allows easy expansion of the stencil library to allow Neumann boundaries to be imposed on the upper or lower row of points of the block.

C.2. Derivation of Computational Stencils

This section provides for each of the 48 possible scenarios,

- the stencil index in 6-bit binary and decimal formats,
- a diagram of the 2×2 block and surrounding points,
- the 5-point operator equations, in matrix form where necessary, and
- the implicit equations for the unknowns within the block

in the format shown in Fig. C-9. The stencils are arranged in order of increasing number of enclosed singularities denoted by the • symbol. For comparative purposes, the equations for the non-accelerated Gauss-Seidel and the successive block over-relaxation (SBOR) methods are provided for the pure Dirichlet boundaries. For reasons of brevity, the solutions involving right or left Neumann boundaries are presented for the Gauss-Seidel block method only: the equations for the SBOR implementation can be obtained from the *sor2x2.cpp* file listed in Appendix F or they can be re-derived by the reader.

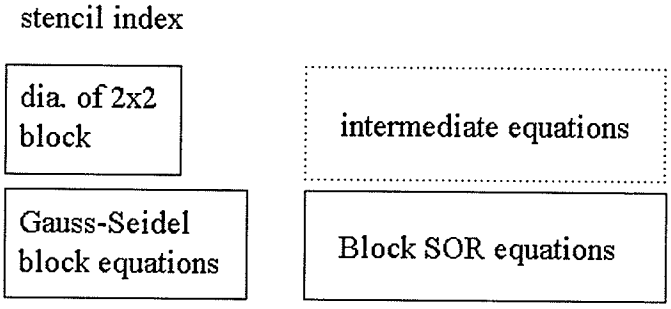


Fig. C - 9: Layout of stencil diagrams and equations.

C.2.1 Pure Dirichlet Boundaries

stencil index: 001111 = 15

$$\begin{array}{cccc} & A & B & \\ C & D & E & F \\ G & H & J & K \\ & L & M & \end{array}$$

$$d = A + C \qquad \Phi = 4(1-\omega)D + \omega(A + C)$$
$$e = B + F \qquad \Lambda = 4(1-\omega)E + \omega(B + F)$$
$$h = G + L \qquad \Psi = 4(1-\omega)H + \omega(G + L)$$
$$j = K + M \qquad \Theta = 4(1-\omega)J + \omega(K + M)$$

$$\begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} d \\ e \\ h \\ j \end{bmatrix} \qquad M = \begin{bmatrix} 4 & -\omega & -\omega & 0 \\ -\omega & 4 & 0 & -\omega \\ -\omega & 0 & 4 & -\omega \\ 0 & -\omega & -\omega & 4 \end{bmatrix}$$

$$D' = \tfrac{1}{24}(7d + 2e + 2h + 1j) \qquad D' = \frac{(8-\omega^2)\Phi + 2\omega\Lambda + 2\omega\Psi + \omega^2\Theta}{8(4-\omega^2)}$$

$$E' = \tfrac{1}{24}(2d + 7e + 1h + 2j) \qquad E' = \frac{2\omega\Phi + (8-\omega^2)\Lambda + \omega^2\Psi + 2\omega\Theta}{8(4-\omega^2)}$$

$$H' = \tfrac{1}{24}(2d + 1e + 7h + 2j) \qquad H' = \frac{2\omega\Phi + \omega^2\Lambda + (8-\omega^2)\Psi + 2\omega\Theta}{8(4-\omega^2)}$$

$$J' = \tfrac{1}{24}(1d + 2e + 2h + 7j) \qquad J' = \frac{\omega^2\Phi + 2\omega\Lambda + 2\omega\Psi + (8-\omega^2)\Theta}{8(4-\omega^2)}$$

stencil index: 001110 = 14

$$\begin{array}{cccc} & A & B & \\ C & D & E & F \\ G & H & \bullet & K \\ & L & M & \end{array}$$

$$d = A + C \qquad \Phi = 4(1-\omega)D + \omega(A + C)$$

$$e = B + F + J \qquad \Lambda = 4(1-\omega)E + \omega(B + F + J)$$

$$h = G + L + J \qquad \Psi = 4(1-\omega)H + \omega(G + L + J)$$

$$\begin{bmatrix} 4 & -1 & -1 \\ -1 & 4 & 0 \\ -1 & 0 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ H' \end{bmatrix} = \begin{bmatrix} d \\ e \\ h \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega & -\omega \\ -\omega & 4 & 0 \\ -\omega & 0 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ H' \end{bmatrix} = \begin{bmatrix} \Phi \\ \Lambda \\ \Psi \end{bmatrix}$$

$$D' = \frac{16\Phi + 4\omega\Lambda + 4\omega\Psi}{8(8-\omega^2)}$$

$$D' = \tfrac{1}{56}(16d + 4e + 4h)$$

$$E' = \tfrac{1}{56}(4d + 15e + 1h) \qquad E' = \frac{4\omega\Phi + (16-\omega^2)\Lambda + \omega^2\Psi}{8(8-\omega^2)}$$

$$H' = \tfrac{1}{56}(4d + 1e + 15h) \qquad H' = \frac{4\omega\Phi + \omega^2\Lambda + (16-\omega^2)\Psi}{8(8-\omega^2)}$$

stencil index: 001101 = 13

$$\begin{array}{cccc} & A & B & \\ C & D & E & F \\ G & \bullet & J & K \\ & L & M & \end{array}$$

$$d = A + C + H \qquad \Phi = 4(1-\omega)D + \omega(A + C + H)$$
$$e = B + F \qquad \Lambda = 4(1-\omega)E + \omega(B + F)$$
$$j = K + M + H \qquad \Psi = 4(1-\omega)J + \omega(K + M + H)$$

$$\begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ J' \end{bmatrix} = \begin{bmatrix} d \\ e \\ j \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega & 0 \\ -\omega & 4 & -\omega \\ 0 & -\omega & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ J' \end{bmatrix} = \begin{bmatrix} \Phi \\ \Lambda \\ \Psi \end{bmatrix}$$

$$D' = \tfrac{1}{56}(15d + 4e + 1j) \qquad D' = \frac{(16 - w^2)\Phi + 4\omega\Lambda + \omega^2\Psi}{8(8 - \omega^2)}$$
$$E' = \tfrac{1}{56}(4d + 16e + 4j) \qquad E' = \frac{4\omega\Phi + 16\Lambda + 4\omega\Psi}{8(8 - \omega^2)}$$
$$J' = \tfrac{1}{56}(1d + 4e + 15j) \qquad J' = \frac{\omega^2\Phi + 4\omega\Lambda + (16 - \omega^2)\Psi}{8(8 - \omega^2)}$$

stencil index: 001011 = 11

$$\begin{array}{cccc} & A & B & \\ C & D & \bullet & F \\ G & H & J & K \\ & L & M & \end{array}$$

$$d = A + C + E \qquad \Phi = 4(1-\omega)D + \omega(A + C + E)$$
$$h = G + L \qquad \Lambda = 4(1-\omega)H + \omega(G + L)$$
$$j = K + M + E \qquad \Psi = 4(1-\omega)J + \omega(K + M + E)$$

$$\begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} D' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} d \\ h \\ j \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega & 0 \\ -\omega & 4 & -\omega \\ 0 & -\omega & 4 \end{bmatrix} \begin{bmatrix} D' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} \Phi \\ \Lambda \\ \Psi \end{bmatrix}$$

$$D' = \tfrac{1}{56}(15d + 4h + 1j)$$
$$H' = \tfrac{1}{56}(4d + 16h + 4j)$$
$$J' = \tfrac{1}{56}(1d + 4h + 15j)$$

$$D' = \frac{(16 - w^2)\Phi + 4\omega\Lambda + \omega^2\Psi}{8(8 - \omega^2)}$$
$$H' = \frac{4\omega\Phi + 16\Lambda + 4\omega\Psi}{8(8 - \omega^2)}$$
$$J' = \frac{\omega^2\Phi + 4\omega\Lambda + (16 - \omega^2)\Psi}{8(8 - \omega^2)}$$

stencil index: 000111 = 7

$$\begin{array}{ccccc} & A & B & & \\ C & \bullet & E & F & \\ G & H & J & K & \\ & L & M & & \end{array}$$

$$e = B + F + D$$
$$h = G + L + D$$
$$j = K + M$$

$$\Phi = 4(1 - \omega)E + \omega(B + F + D)$$
$$\Lambda = 4(1 - \omega)H + \omega(G + L + D)$$
$$\Psi = 4(1 - \omega)J + \omega(K + M)$$

$$\begin{bmatrix} 4 & 0 & -1 \\ 0 & 4 & -1 \\ -1 & -1 & 4 \end{bmatrix} \begin{bmatrix} E' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} e \\ h \\ j \end{bmatrix}$$

$$\begin{bmatrix} 4 & 0 & -\omega \\ 0 & 4 & -\omega \\ -\omega & -\omega & 4 \end{bmatrix} \begin{bmatrix} E' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} \Phi \\ \Lambda \\ \Psi \end{bmatrix}$$

$$E' = \tfrac{1}{56}(15e + 1h + 4j)$$
$$H' = \tfrac{1}{56}(1e + 15h + 4j)$$
$$J' = \tfrac{1}{56}(4e + 4h + 16j)$$

$$E' = \frac{(16 - w^2)\Phi + \omega^2\Lambda + 4\omega\Psi}{8(8 - \omega^2)}$$
$$H' = \frac{\omega^2\Phi + (16 - w^2)\Lambda + 4\omega\Psi}{8(8 - \omega^2)}$$
$$J' = \frac{4\omega\Phi + 4\omega\Lambda + 16\Psi}{8(8 - \omega^2)}$$

stencil index: 001100 = 12

$$\begin{array}{ccccc} & A & B & & \\ C & D & E & F & \\ G & \bullet & \bullet & K & \\ & L & M & & \end{array}$$

$$d = A + C + H \qquad \Phi = \omega(A + C + H) + 4(1-\omega)D$$
$$e = B + F + J \qquad \Lambda = \omega(B + F + J) + 4(1-\omega)E$$

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} D' \\ E' \end{bmatrix} = \begin{bmatrix} d \\ e \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega \\ -\omega & 4 \end{bmatrix}\begin{bmatrix} D' \\ E' \end{bmatrix} = \begin{bmatrix} \Phi \\ \Lambda \end{bmatrix}$$

$$D' = 1/15(4d + 1e) \qquad D' = \frac{4\Phi + \omega\Lambda}{16 - w^2}, \quad E' = \frac{\omega\Phi + 4\Lambda}{16 - w^2}$$
$$E' = 1/15(1d + 4e)$$

stencil index: 000011 = 3

$$\begin{array}{cccc} & A & B & \\ C & \bullet & \bullet & F \\ G & H & J & K \\ & L & M & \end{array}$$

$$h = G + L + D \qquad \Phi = \omega(G + L + D) + 4(1-\omega)H$$
$$j = K + M + E \qquad \Lambda = \omega(K + M + E) + 4(1-\omega)J$$

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} H' \\ J' \end{bmatrix} = \begin{bmatrix} h \\ j \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega \\ -\omega & 4 \end{bmatrix}\begin{bmatrix} H' \\ J' \end{bmatrix} = \begin{bmatrix} \Phi \\ \Lambda \end{bmatrix}$$

$$H' = 1/15(4h + j) \qquad H' = \frac{4\Phi + \omega\Lambda}{16 - w^2}, \quad J' = \frac{\omega\Phi + 4\Lambda}{16 - w^2}$$
$$J' = 1/15(h + 4j)$$

stencil index: 001010 = 10

$$\begin{array}{cccc} & A & B & \\ C & D & \bullet & F \\ G & H & \bullet & K \\ & L & M & \end{array}$$

$$d = A + C + E \qquad \Phi = \omega(A + C + E) + 4(1-\omega)D$$
$$h = G + L + J \qquad \Lambda = \omega(G + L + J) + 4(1-\omega)H$$

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} D' \\ H' \end{bmatrix} = \begin{bmatrix} d \\ h \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega \\ -\omega & 4 \end{bmatrix}\begin{bmatrix} D' \\ H' \end{bmatrix} = \begin{bmatrix} \Phi \\ \Lambda \end{bmatrix}$$

$$D' = 1/15(4d + h)$$
$$H' = 1/15(d + 4h)$$

$$D' = \frac{4\Phi + \omega\Lambda}{16 - w^2}, \quad H' = \frac{\omega\Phi + 4\Lambda}{16 - w^2}$$

stencil index: 000101 = 5

$$\begin{array}{cccc} & A & B & \\ C & \bullet & E & F \\ G & \bullet & J & K \\ & L & M & \end{array}$$

$$e = B + F + D$$
$$j = K + M + H$$

$$\Phi = \omega(B + F + D) + 4(1 - \omega)E$$
$$\Lambda = \omega(K + M + H) + 4(1 - \omega)J$$

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} E' \\ J' \end{bmatrix} = \begin{bmatrix} e \\ j \end{bmatrix}$$

$$\begin{bmatrix} 4 & -\omega \\ -\omega & 4 \end{bmatrix}\begin{bmatrix} E' \\ J' \end{bmatrix} = \begin{bmatrix} \Phi \\ \Lambda \end{bmatrix}$$

$$E' = 1/15(4e + j)$$
$$J' = 1/15(e + 4j)$$

$$E' = \frac{4\Phi + \omega\Lambda}{16 - w^2}, \quad J' = \frac{\omega\Phi + 4\Lambda}{16 - w^2}$$

stencil index: 001001 = 9

$$\begin{array}{cccc} & A & B & \\ C & D & \bullet & F \\ G & \bullet & J & K \\ & L & M & \end{array}$$

$$D' = 1/4(A + C + E + H)$$
$$J' = 1/4(K + M + E + H)$$

$$D' = \frac{\omega}{4}(A + C + E + H) + (1 - \omega)D$$
$$J' = \frac{\omega}{4}(K + M + E + H) + (1 - \omega)J$$

stencil index: 000110 = 6

$$\begin{array}{cccc} & A & B & \\ C & \bullet & E & F \\ G & H & \bullet & K \\ & L & M & \end{array}$$

$$E' = \tfrac{1}{4}(B + F + D + J) \qquad E' = \frac{\omega}{4}(B + F + D + J) + (1 - \omega)E$$
$$H' = \tfrac{1}{4}(G + L + D + J) \qquad H' = \frac{\omega}{4}(G + L + D + J) + (1 - \omega)H$$

stencil index: 001000 = 8

$$\begin{array}{cccc} & A & B & \\ C & D & \bullet & F \\ G & \bullet & \bullet & K \\ & L & M & \end{array}$$

$$D' = \tfrac{1}{4}(A + C + E + H) \qquad D' = \frac{\omega}{4}(A + C + E + H) + (1 - \omega)D$$

stencil index: 000100 = 4

$$\begin{array}{cccc} & A & B & \\ C & \bullet & E & F \\ G & \bullet & \bullet & K \\ & L & M & \end{array}$$

$$E' = \tfrac{1}{4}(B + F + D + J) \qquad E' = \frac{\omega}{4}(B + F + D + J) + (1 - \omega)E$$

stencil index: 000010 = 2

$$\begin{array}{cccc} & A & B & \\ C & \bullet & \bullet & F \\ G & H & \bullet & K \\ & L & M & \end{array}$$

$$H' = \tfrac{1}{4}(G + L + D + J) \qquad H' = \frac{\omega}{4}(G + L + D + J) + (1 - \omega)H$$

stencil index: 000001 = 1

$$\begin{array}{cccc} & A & B & \\ C & \bullet & \bullet & F \\ G & \bullet & J & K \\ & L & M & \end{array}$$

$$J' = \tfrac{1}{4}(K + M + E + H) \qquad J' = \frac{\omega}{4}(K + M + E + H) + (1 - \omega)J$$

stencil index: 000000 = 0

$$\begin{array}{cccc} & A & B & \\ C & \bullet & \bullet & F \\ G & \bullet & \bullet & K \\ & L & M & \end{array}$$

no computations required as all values of block are known

C.2.2 Neumann Boundaries Along the Right Side

stencil index: 011111 = 31

$$\begin{array}{cccc} & A & B & \\ C & D & E & \\ G & H & J & \\ & L & M & \end{array} \qquad \begin{aligned} d &= A + C \\ e &= B \\ h &= G + L \\ j &= M \end{aligned}$$

$$\begin{bmatrix} 4 & -1 & -1 & 0 \\ -2 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -2 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} d \\ e \\ h \\ j \end{bmatrix} \qquad \begin{aligned} D' &= \tfrac{1}{161}(52d + 15e + 17h + 8j) \\ E' &= \tfrac{1}{161}(30d + 52e + 16h + 17j) \\ H' &= \tfrac{1}{161}(17d + 8e + 52h + 15j) \\ J' &= \tfrac{1}{161}(16d + 17e + 30h + 52j) \end{aligned}$$

stencil index: 011110 = 30

$$\begin{array}{ccc} & A & B \\ C & D & E \\ G & H & \bullet \\ & L & M \end{array}$$

$$\begin{aligned} d &= A + C \\ e &= B + J \\ h &= G + L + J \end{aligned} \qquad \begin{bmatrix} 4 & -1 & -1 \\ -2 & 4 & 0 \\ -1 & 0 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ H' \end{bmatrix} = \begin{bmatrix} d \\ e \\ h \end{bmatrix} \qquad \begin{aligned} D' &= \tfrac{1}{52}(16d + 4e + 4h) \\ E' &= \tfrac{1}{52}(8d + 15e + 2h) \\ H' &= \tfrac{1}{52}(4d + 1e + 14h) \end{aligned}$$

stencil index: 011101 = 29

$$\begin{array}{ccc} & A & B \\ C & D & E \\ G & \bullet & J \\ & L & M \end{array}$$

$$\begin{aligned} d &= A + C + H \\ e &= B \\ j &= M + 2H \end{aligned} \qquad \begin{bmatrix} 4 & -1 & 0 \\ -2 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ J' \end{bmatrix} = \begin{bmatrix} d \\ e \\ j \end{bmatrix} \qquad \begin{aligned} D' &= \tfrac{1}{52}(15d + 4e + 1j) \\ E' &= \tfrac{1}{52}(8d + 16e + 4j) \\ J' &= \tfrac{1}{52}(2d + 4e + 14j) \end{aligned}$$

stencil index: 011011 = 27

$$\begin{array}{ccc} & A & B \\ C & D & \bullet \\ G & H & J \\ & L & M \end{array}$$

$$\begin{aligned} d &= A + C + E \\ h &= G + L \\ j &= M + E \end{aligned} \qquad \begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -2 & 4 \end{bmatrix} \begin{bmatrix} D' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} d \\ h \\ j \end{bmatrix} \qquad \begin{aligned} D' &= \tfrac{1}{52}(14d + 4h + 1j) \\ H' &= \tfrac{1}{52}(4d + 16h + 4j) \\ J' &= \tfrac{1}{52}(2d + 8h + 15j) \end{aligned}$$

stencil index: 010111 = 23

$$\begin{array}{ccc} & A & B \\ C & \bullet & E \\ G & H & J \\ & L & M \end{array}$$

$$\begin{aligned} e &= B + 2D \\ h &= G + L + D \\ j &= M \end{aligned} \qquad \begin{bmatrix} 4 & 0 & -1 \\ 0 & 4 & -1 \\ -1 & -2 & 4 \end{bmatrix} \begin{bmatrix} E' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} e \\ h \\ j \end{bmatrix} \qquad \begin{aligned} E' &= 1/52\,(14e + 2h + 4j) \\ H' &= 1/52\,(1e + 15h + 4j) \\ J' &= 1/52\,(4e + 8h + 16j) \end{aligned}$$

stencil index: 011100 = 28

$$\begin{array}{ccc} & A & B \\ C & D & E \\ G & \bullet & \bullet \\ & L & M \end{array}$$

$$\begin{aligned} d &= A + C + H \\ e &= B + J \end{aligned} \qquad \begin{bmatrix} 4 & -1 \\ -2 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \end{bmatrix} = \begin{bmatrix} d \\ e \end{bmatrix} \qquad \begin{aligned} D' &= 1/14\,(4d + 1e) \\ E' &= 1/14\,(2d + 4e) \end{aligned}$$

stencil index: 010011 = 19

$$\begin{array}{ccc} & A & B \\ C & \bullet & \bullet \\ G & H & J \\ & L & M \end{array}$$

$$\begin{aligned} h &= G + L + D \\ j &= M + E \end{aligned} \qquad \begin{bmatrix} 4 & -1 \\ -2 & 4 \end{bmatrix} \begin{bmatrix} H' \\ J' \end{bmatrix} = \begin{bmatrix} h \\ j \end{bmatrix} \qquad \begin{aligned} H' &= 1/14\,(4h + 1j) \\ J' &= 1/14\,(2h + 4j) \end{aligned}$$

stencil index: 011010 = 26

$$\begin{array}{ccc} & A & B \\ C & D & \bullet \\ G & H & \bullet \\ & L & M \end{array}$$

$$d = A + C + E \qquad \begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix} \begin{bmatrix} D' \\ H' \end{bmatrix} = \begin{bmatrix} d \\ h \end{bmatrix} \qquad D' = \tfrac{1}{15}(4d + h)$$
$$h = G + L + J \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad H' = \tfrac{1}{15}(d + 4h)$$

stencil index: 010101 = 21

$$\begin{array}{ccc} & A & B \\ C & \bullet & E \\ G & \bullet & J \\ & L & M \end{array}$$

$$e = B + 2D \qquad \begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix} \begin{bmatrix} E' \\ J' \end{bmatrix} = \begin{bmatrix} e \\ j \end{bmatrix} \qquad E' = \tfrac{1}{15}(4e + j)$$
$$j = M + 2H \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad J' = \tfrac{1}{15}(e + 4j)$$

stencil index: 011001 = 25

$$\begin{array}{ccc} & A & B \\ C & D & \bullet \\ G & \bullet & J \\ & L & M \end{array}$$

$$D' = \tfrac{1}{4}(A + C + E + H)$$
$$J' = \tfrac{1}{4}(M + E + 2H)$$

stencil index: 010110 = 22

$$\begin{array}{ccc} & A & B \\ C & \bullet & E \\ G & H & \bullet \\ & L & M \end{array}$$

$$E' = \tfrac{1}{4}(B + 2D + J)$$
$$H' = \tfrac{1}{4}(G + L + D + J)$$

stencil index: 011000 = 24

$$\begin{array}{ccc} & A & B \\ C & D & \bullet \\ G & \bullet & \bullet \\ & L & M \end{array}$$

$$D' = \tfrac{1}{4}(A + C + E + H)$$

stencil index: 010100 = 20

$$\begin{array}{ccc} & A & B \\ C & \bullet & E \\ G & \bullet & \bullet \\ & L & M \end{array}$$

$$E' = \tfrac{1}{4}(B + 2D + J)$$

stencil index: 010010 = 18

$$\begin{array}{ccc} & A & B \\ C & \bullet & \bullet \\ G & H & \bullet \\ & L & M \end{array}$$

$$H' = \tfrac{1}{4}(G + L + D + J)$$

stencil index: 010001 = 17

$$\begin{array}{ccc} & A & B \\ C & \bullet & \bullet \\ G & \bullet & J \\ & L & M \end{array}$$

$$J' = \tfrac{1}{4}(M + E + 2H)$$

stencil index: 010000 = 16

$$\begin{array}{ccc} & A & B \\ C & \bullet & \bullet \\ G & \bullet & \bullet \\ & L & M \end{array}$$

no computations required as all values of block are known

C.2.3 Neumann Boundaries Along the Left Side

stencil index: 101111 = 47

$$\begin{array}{ccc} A & B & \\ D & E & F \\ H & J & K \\ L & M & \end{array} \qquad \begin{aligned} d &= A \\ e &= B + F \\ h &= L \\ j &= K + M \end{aligned}$$

$$\begin{bmatrix} 4 & -2 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -2 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} d \\ e \\ h \\ j \end{bmatrix} \qquad \begin{aligned} D' &= \tfrac{1}{161}(52d + 30e + 17h + 16j) \\ E' &= \tfrac{1}{161}(15d + 52e + 8h + 17j) \\ H' &= \tfrac{1}{161}(17d + 16e + 52h + 30j) \\ J' &= \tfrac{1}{161}(8d + 17e + 15h + 52j) \end{aligned}$$

stencil index: 101110 = 46

$$\begin{array}{ccc} A & B & \\ D & E & F \\ H & \bullet & K \\ L & M & \end{array}$$

$$\begin{aligned} d &= A \\ e &= B + F + J \\ h &= L + 2J \end{aligned} \qquad \begin{bmatrix} 4 & -2 & -1 \\ -1 & 4 & 0 \\ -1 & 0 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ H' \end{bmatrix} = \begin{bmatrix} d \\ e \\ h \end{bmatrix} \qquad \begin{aligned} D' &= \tfrac{1}{52}(16d + 8e + 4h) \\ E' &= \tfrac{1}{52}(4d + 15e + 1h) \\ H' &= \tfrac{1}{52}(4d + 2e + 14h) \end{aligned}$$

stencil index: 101101 = 45

$$\begin{array}{ccc} A & B & \\ D & E & F \\ \bullet & J & K \\ L & M & \end{array}$$

$$\begin{aligned} d &= A + H \\ e &= B + F \\ j &= K + M + H \end{aligned} \qquad \begin{bmatrix} 4 & -2 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} D' \\ E' \\ J' \end{bmatrix} = \begin{bmatrix} d \\ e \\ j \end{bmatrix} \qquad \begin{aligned} D' &= 1/52\,(15d + 8e + 2j) \\ E' &= 1/52\,(4d + 16e + 4j) \\ J' &= 1/52\,(1d + 4e + 14j) \end{aligned}$$

stencil index: 101011 = 43

$$\begin{array}{ccc} A & B & \\ D & \bullet & F \\ H & J & K \\ L & M & \end{array}$$

$$\begin{aligned} d &= A + 2E \\ h &= L \\ j &= K + M + E \end{aligned} \qquad \begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -2 \\ 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} D' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} d \\ h \\ j \end{bmatrix} \qquad \begin{aligned} D' &= 1/52\,(14d + 4h + 2j) \\ H' &= 1/52\,(4d + 16h + 8j) \\ J' &= 1/52\,(1d + 4h + 15j) \end{aligned}$$

stencil index: 100111 = 39

$$\begin{array}{ccc} A & B & \\ \bullet & E & F \\ H & J & K \\ L & M & \end{array}$$

$$\begin{aligned} e &= B + F + D \\ h &= L + D \\ j &= K + M \end{aligned} \qquad \begin{bmatrix} 4 & 0 & -1 \\ 0 & 4 & -2 \\ -1 & -1 & 4 \end{bmatrix} \begin{bmatrix} E' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} e \\ h \\ j \end{bmatrix} \qquad \begin{aligned} E' &= 1/52\,(14e + 1h + 4j) \\ H' &= 1/52\,(2e + 15h + 8j) \\ J' &= 1/52\,(4e + 4h + 16j) \end{aligned}$$

stencil index: 101100 = 44

$$\begin{array}{ccc} A & B & \\ D & E & F \\ \bullet & \bullet & K \\ L & M & \end{array}$$

$$d = A + H \qquad \begin{bmatrix} 4 & -2 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} D' \\ E' \end{bmatrix} = \begin{bmatrix} d \\ e \end{bmatrix} \qquad D' = \tfrac{1}{14}(4d + 2e)$$
$$e = B + F + J \qquad\qquad\qquad\qquad\qquad\qquad E' = \tfrac{1}{14}(1d + 4e)$$

stencil index: 100011 = 35

$$\begin{array}{ccc} A & B & \\ \bullet & \bullet & F \\ H & J & K \\ L & M & \end{array}$$

$$h = L + D \qquad \begin{bmatrix} 4 & -2 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} H' \\ J' \end{bmatrix} = \begin{bmatrix} h \\ j \end{bmatrix} \qquad H' = \tfrac{1}{14}(4h + 2j)$$
$$j = K + M + E \qquad\qquad\qquad\qquad\qquad\qquad J' = \tfrac{1}{14}(1h + 4j)$$

stencil index: 101010 = 42

$$\begin{array}{ccc} A & B & \\ D & \bullet & F \\ H & \bullet & K \\ L & M & \end{array}$$

$$d = A + 2E \qquad \begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} D' \\ H' \end{bmatrix} = \begin{bmatrix} d \\ h \end{bmatrix} \qquad D' = \tfrac{1}{15}(4d + h)$$
$$h = L + 2J \qquad\qquad\qquad\qquad\qquad\qquad H' = \tfrac{1}{15}(d + 4h)$$

stencil index: 100101 = 37

$$\begin{array}{ccc} A & B & \\ \bullet & E & F \\ \bullet & J & K \\ L & M & \end{array}$$

$$e = B + F + D \qquad j = K + M + H$$

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix} \begin{bmatrix} E' \\ J' \end{bmatrix} = \begin{bmatrix} e \\ j \end{bmatrix}$$

$$E' = \tfrac{1}{15}(4e + j)$$
$$J' = \tfrac{1}{15}(e + 4j)$$

stencil index: 101001 = 41

$A \quad B$
$D \quad \bullet \quad F$
$\bullet \quad J \quad K$
$L \quad M$

$D' = \tfrac{1}{4}(A + 2E + H)$
$J' = \tfrac{1}{4}(K + M + E + H)$

stencil index: 100110 = 38

$A \quad B$
$\bullet \quad E \quad F$
$H \quad \bullet \quad K$
$L \quad M$

$E' = \tfrac{1}{4}(B + F + D + J)$
$H' = \tfrac{1}{4}(L + D + 2J)$

stencil index: 101000 = 40

$A \quad B$
$D \quad \bullet \quad F$
$\bullet \quad \bullet \quad K$
$L \quad M$

$D' = \tfrac{1}{4}(A + 2E + H)$

stencil index: 100100 = 36

$A \quad B$
$\bullet \quad E \quad F$
$\bullet \quad \bullet \quad K$
$L \quad M$

$$E' = \tfrac{1}{4}(B + F + D + J)$$

stencil index: 100010 = 34

$A \quad B$
$\bullet \quad \bullet \quad F$
$H \quad \bullet \quad K$
$L \quad M$

$$H' = \tfrac{1}{4}(L + D + 2J)$$

stencil index: 100001 = 33

$A \quad B$
$\bullet \quad \bullet \quad F$
$\bullet \quad J \quad K$
$L \quad M$

$$J' = \tfrac{1}{4}(K + M + E + H)$$

stencil index: 100000 = 32

$A \quad B$
$\bullet \quad \bullet \quad F$
$\bullet \quad \bullet \quad K$
$L \quad M$

no computations required as all values of block are known

C.3. Extensions to Larger Blocks

The next largest block size after 2×2 blocks is 3×3 blocks. If left and right Neumann boundaries and enclosed singularities are to be considered, then for every 3×3 block that is evaluated,

- 9 mesh points contribute to the stencil index instead of 4,

- 9 coefficients must be pre-calculated instead of 4 (d, e, h, j),
- upto 9 terms instead of 4 must be summed together to update a given potential, and
- the denominators are much more complicated, involving more computations.

The ultimate prohibiting factor is that a total of 1536 different scenarios must be handled. Stencil rotation is too involved, a generalized inverse matrix with Kroniker delta-type multipliers should not even be considered, and coding 1536 unique non-rotatable stencil is too error prone and requires massive amounts of code that would need to be debugged and compiled.

If larger stencils are used, it is recommended that they only be used on singular-free mesh regions. If Neumann boundaries are supported, then only 3 stencils would be required for any block size, $n \times n$.

APPENDIX D

ITERATIVE LINE METHODS

D.1. Introduction

This appendix deals with the implementation of line methods for the iterative solution of Laplace's equation on a discrete mesh. Line iterative methods iterate the mesh points, line-by-line, simultaneously solving for all points on a given line. The lines of points to be solved may be horizontal rows or vertical columns of points.

D.1.1 Restrictions on Boundaries

For the purposes of this thesis, the scope of the research has been restricted to the modeling of Laplacian fractals within the following configurations:

- parallel plates,
- coaxial with centre point or disc electrode,
- rectangular with centre disc electrode.

As a direct result of our restricted set of configurations, the topmost and bottommost mesh rows are ***always*** Dirichlet boundaries. Additionally, the leftmost and rightmost columns are always Dirichlet boundaries, except for the parallel plates configuration in which they both become Neumann boundaries. Refer to Chapter 2 (Sec. 2.4.2) for a discussion of singularities and Dirichlet and Neumann boundaries.

D.1.2 Complications Due to Enclosed Singularities

The existence of singularities in a line of points necessitates the segmentation of that line into singularity-free *line segments*, each segment terminated by Dirichlet or Neumann boundaries. This must be done since the solution of that line cannot modify the values of the singular points.

Writing out the 5-point operator equations for all points along a *line segment* or a singularity-free *line* results in a tridiagonal matrix of coefficients which can be easily inverted. The potentials for points on short line segments are quickly solved using precalculated matrix inverses, the potentials for points on longer line segments are solved using tridiagonal algorithms.

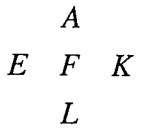
In contrast, point iterative methods deal with singularities by "skipping" over them, block iterative methods handle enclosed singularities by using a library of unique stencils, and line iterative methods handle enclosed singularities by segmentation of the line into singularity-free line segments.

The implementation of the simultaneous solution of 2 adjacent lines of points, Successive 2-Line Over-Relaxation (S2LOR), is not recommended due to the severe non-aligned segmentation caused by non-aligned, scattered singularities which may occur throughout the 2 lines to be solved.

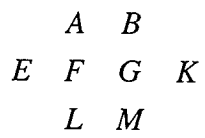
D.1.3 Conventions for Short Line Segments

Figure D-1 shows the labeling convention used hereafter for short row segments consisting of 1, 2, 3, and 4 mesh points at the k^{th} stage of iteration, with unknown potentials represented by F, G, H, and J, (dependent on the segment size), and having

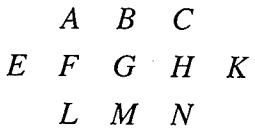
adjacent points with potentials of A, B, C, D, L, M, N, and P. It is desired to find the solution A', B', C', and D' at the iteration stage k+1. The letters A through M are used for clarity and brevity instead of the longer double-scripted $U_{i,j}^{k}$ notation, and the letters A', B', C', and D' are used instead of the $U_{i,j}^{k+1}$ notation in Chapter 2.



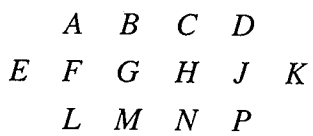
a. segment with 1 point (F)



b. segment with 2 points (F,G)



c. segment with 3 points (F,G,H)



d. segment with 4 points (F,G,H,J)

Fig. D - 1: Examples of short segments.

This row segment labeling scheme can be rotated 90° clockwise and applied to column segments.

The remainder of this appendix is dedicated to formulas required to solve for rows and columns of points using line iterative methods. For each of the 4 possible segment boundary conditions, the generalized matrix of 5-point operator coefficients is provided, and for small segments of less than 5 points, provided are:

- a labeled diagram of the line segment and surrounding points,
- the 5-point operator equations, in matrix form where warranted, and
- the implicit equations for the unknown potentials of points constituting the segment.

Singularities at segment endpoints are represented by $\bullet_E$ and $\bullet_K$, and points with imposed Neumann boundaries are indicated by surrounding [ ] brackets, such as [A]. For

comparative purposes, the equations for the non-accelerated Gauss-Seidel method (GLOR or GSLOR) appear in the left half of the pages, and the equations for the accelerated successive line over-relaxation (SLOR) methods appear on the right half of the pages

D.2. Horizontal Lines

The solution of horizontal lines, or rows, of points is the orientation most commonly associated with line methods.

D.2.1 Typical Rows of Points

Parallel plate capacitors are modeled as having Dirichlet boundaries along the plates, and Neumann boundaries along the leftmost and rightmost columns. Figure D-2 shows several horizontal rows of points from a discrete, mesh-based representation of a parallel plate capacitor. The upper highlighted row is singularity-free and has Neumann boundaries imposed upon both its endpoints. The second highlighted row contains 2 enclosed singularities due to the corona branches "cutting" or crossing that row. The properly segmented row consists of 3 line segments, such that:

- the left segment contains the left Neumann boundary, and a right Dirichlet boundary due to the leftmost singularity,
- the middle segment has Dirichlet boundaries due to the singularities at its endpoints, and
- the right segment contains the right Neumann boundary, and a left Dirichlet boundary due to the rightmost singularity.

Thus, for rows of points, we have 4 possible combinations of boundaries imposed upon the segment endpoints. During the solution of any given row, *both adjacent rows* are treated as temporary Dirichlet boundaries.

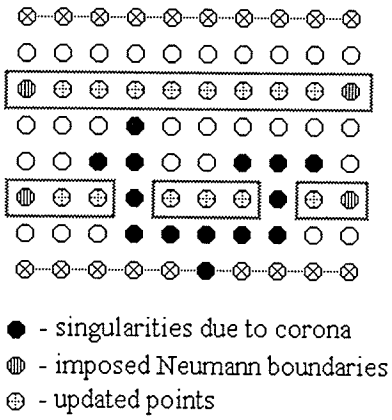


Fig. D - 2: Typical rows of mesh points.

D.2.2 Solution Approaches

Writing out the 5-point operator equations for all points along a line segment or singular-free line results in a tridiagonal matrix which can be easily inverted. The potentials for points on long line segments, consisting of at least 5 points, are solved using tridiagonal algorithms. The potentials for points on short line segments, consisting of less than 5 points, are quickly solved using precalculated matrix inverses. In areas of dense fractal growth, many very short line segments exist, and a fast solution method is essential for computation speed.

The following sections detail the general solution for long row segments and the direct solutions for short row segments for the four different boundary conditions.

D.2.3 Pure Dirichlet Boundaries

D.2.3.1 General Matrix Form

It can be shown that the generalized matrix M of coefficients of the 5-point operator equations has the form

$$M = \begin{bmatrix} 4 & -1 & 0 & 0 & 0 \\ -1 & 4 & \ddots & 0 & 0 \\ 0 & \ddots & \ddots & -1 & 0 \\ 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & -1 & 4 \end{bmatrix}. \qquad \text{(D-1)}$$

D.2.3.2 Direct Solution of Short Segments

1-point row segments:

$$\begin{array}{ccc} & A & \\ \bullet_E & F & \bullet_K \\ & L & \end{array}$$

$$F' = \tfrac{1}{4}(A + L + \bullet_E + \bullet_K) \qquad F' = \frac{\omega}{4}(A + L + \bullet_E + \bullet_K) + (1-\omega)F$$

2-point row segments:

$$\begin{array}{ccc} A & B & \\ \bullet_E & F \quad G & \bullet_K \\ L & M & \end{array}$$

$$f = A + L + \bullet_E \qquad \phi = 4(1-\omega)F + \omega(A + L + \bullet_E)$$
$$g = B + M + \bullet_K \qquad \gamma = 4(1-\omega)G + \omega(B + M + \bullet_K)$$

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega \\ -\omega & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \end{bmatrix}$$

$$F' = \tfrac{1}{15}(4f + g)$$
$$G' = \tfrac{1}{15}(f + 4g)$$

$$F' = \frac{4\phi + \omega\gamma}{16 - \omega^2}$$
$$G' = \frac{\omega\phi + 4\gamma}{16 - \omega^2}$$

3-point row segments:

$$\begin{array}{cccc} & A & B & C & \\ \bullet_E & F & G & H & \bullet_K \\ & L & M & N & \end{array}$$

$$f = A + L + \bullet_E$$
$$g = B + M$$
$$h = C + N + \bullet_K$$

$$\begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \end{bmatrix}$$

$$F' = \tfrac{1}{56}(15f + 4g + 1h)$$
$$G' = \tfrac{1}{56}(4f + 16g + 4h)$$
$$H' = \tfrac{1}{56}(1f + 4g + 15h)$$

$$\phi = 4(1-\omega)F + \omega(A + L + \bullet_E)$$
$$\gamma = 4(1-\omega)G + \omega(B + M)$$
$$\eta = 4(1-\omega)H + \omega(C + N + \bullet_K)$$

$$\begin{bmatrix} 4 & -\omega & 0 \\ -\omega & 4 & -\omega \\ 0 & -\omega & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \end{bmatrix}$$

$$F' = \frac{(16 - \omega^2)\phi + 4\omega\gamma + \omega^2\eta}{8(8 - \omega^2)}$$
$$G' = \frac{4\omega\phi + 16\gamma + 4\omega\eta}{8(8 - \omega^2)}$$
$$H' = \frac{\omega^2\phi + 4\omega\gamma + (16 - \omega^2)\eta}{8(8 - \omega^2)}$$

4-point row segments:

$$\begin{array}{ccccc} & A & B & C & D & \\ \bullet_E & F & G & H & J & \bullet_K \\ & L & M & N & P & \end{array}$$

$$f = A + L + \bullet_E$$
$$g = B + M$$
$$h = C + N$$
$$j = D + P + \bullet_K$$

$$\phi = 4(1-\omega)F + \omega(A + L + \bullet_E)$$
$$\gamma = 4(1-\omega)G + \omega(B + M)$$
$$\eta = 4(1-\omega)H + \omega(C + N)$$
$$\lambda = 4(1-\omega)J + \omega(D + P + \bullet_K)$$

$$\begin{bmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \\ j \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega & 0 & 0 \\ -\omega & 4 & -\omega & 0 \\ 0 & -\omega & 4 & -\omega \\ 0 & 0 & -\omega & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \\ \lambda \end{bmatrix}$$

$$F' = \tfrac{1}{209}(56f + 15g + 4h + 1j)$$
$$G' = \tfrac{1}{209}(15f + 60g + 16h + 4j)$$
$$H' = \tfrac{1}{209}(4f + 16g + 60h + 15j)$$
$$J' = \tfrac{1}{209}(1f + 4g + 15h + 56j)$$

[see D.4.]

D.2.4 Left Side Neumann Boundaries

D.2.4.1 General Matrix Form

It can be shown that the generalized matrix M of coefficients of the 5-point operator equations has the form

$$M = \begin{bmatrix} 4 & -2 & 0 & 0 & 0 \\ -1 & 4 & \ddots & 0 & 0 \\ 0 & \ddots & \ddots & -1 & 0 \\ 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & -1 & 4 \end{bmatrix}. \qquad \text{(D-2)}$$

D.2.4.2 Direct Solution of Short Segments

1-point row segments:

$$\begin{matrix} A \\ [F] \bullet_K \\ L \end{matrix}$$

$$F' = \tfrac{1}{4}(A + L + 2\bullet_K) \qquad F' = (1-\omega)F + \frac{\omega}{4}(A + L + 2\bullet_K)$$

2-point row segments:

$$\begin{matrix} A & B & \\ [F] & G & \bullet_K \\ L & M & \end{matrix}$$

$$f = A + L$$
$$g = B + M + \bullet_K$$

$$\begin{bmatrix} 4 & -2 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}$$

$$F' = \tfrac{1}{14}(4f + 2g)$$
$$G' = \tfrac{1}{14}(f + 4g)$$

$$\phi = 4(1-\omega)F + \omega(A + L)$$
$$\gamma = 4(1-\omega)G + \omega(B + M + \bullet_K)$$

$$\begin{bmatrix} 4 & -2\omega \\ -\omega & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \end{bmatrix}$$

$$F' = \frac{4\phi + 2\omega\gamma}{16 - 2\omega^2}$$
$$G' = \frac{\omega\phi + 4\gamma}{16 - 2\omega^2}$$

3-point row segments:

$$\begin{matrix} A & B & C & \\ [F] & G & H & \bullet_K \\ L & M & N & \end{matrix}$$

$$f = A + L$$
$$g = B + M$$
$$h = C + N + \bullet_K$$

$$\begin{bmatrix} 4 & -2 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \end{bmatrix}$$

$$F' = \tfrac{1}{52}(15f + 8g + 2h)$$
$$G' = \tfrac{1}{52}(4f + 16g + 4h)$$
$$H' = \tfrac{1}{52}(1f + 4g + 14h)$$

$$\phi = 4(1-\omega)F + \omega(A + L)$$
$$\gamma = 4(1-\omega)G + \omega(B + M)$$
$$\eta = 4(1-\omega)H + \omega(C + N + \bullet_K)$$

$$\begin{bmatrix} 4 & -2\omega & 0 \\ -\omega & 4 & -\omega \\ 0 & -\omega & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \end{bmatrix}$$

$$F' = \frac{(16 - \omega^2)\phi + 8\omega\gamma + 2\omega^2\eta}{4(16 - 3\omega^2)}$$
$$G' = \frac{4\omega\phi + 16\gamma + 4\omega\eta}{4(16 - 3\omega^2)}$$
$$H' = \frac{\omega^2\phi + 4\omega\gamma + (16 - 2\omega^2)\eta}{4(16 - 3\omega^2)}$$

4-point row segments:

$$\begin{matrix} A & B & C & D \\ [F] & G & H & J & \bullet_K \\ L & M & N & P \end{matrix}$$

$$f = A + L$$
$$g = B + M$$
$$h = C + N$$
$$j = D + P + \bullet_K$$

$$\begin{bmatrix} 4 & -2 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \\ j \end{bmatrix}$$

$$F' = 1/194\,(56f + 30g + 8h + 2j)$$
$$G' = 1/194\,(15f + 60g + 16h + 4j)$$
$$H' = 1/194\,(4f + 16g + 56h + 14j)$$
$$J' = 1/194\,(1f + 4g + 14h + 52j)$$

$$\phi = 4(1-\omega)F + \omega(A + L)$$
$$\gamma = 4(1-\omega)G + \omega(B + M)$$
$$\eta = 4(1-\omega)H + \omega(C + N)$$
$$\lambda = 4(1-\omega)J + \omega(D + P + \bullet_K)$$

$$\begin{bmatrix} 4 & -2\omega & 0 & 0 \\ -\omega & 4 & -\omega & 0 \\ 0 & -\omega & 4 & -\omega \\ 0 & 0 & -\omega & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \\ \lambda \end{bmatrix}$$

[see D.4.]

D.2.5 Right Side Neumann Boundaries

D.2.5.1 General Matrix Form

It can be shown that the generalized matrix M of coefficients of the 5-point operator equations has the form

$$M = \begin{bmatrix} 4 & -1 & 0 & 0 & 0 \\ -1 & 4 & \ddots & 0 & 0 \\ 0 & \ddots & \ddots & -1 & 0 \\ 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & -2 & 4 \end{bmatrix}. \qquad \text{(D-3)}$$

D.2.5.2 Direct Solution of Short Segments

1-point row segments:

$$\begin{matrix} & A \\ \bullet_E & [F] \\ & L \end{matrix}$$

$$F' = \tfrac{1}{4}(A + L + 2\bullet_E)$$

$$F' = (1-\omega)F + \frac{\omega}{4}(A + L + 2\bullet_E)$$

2-point row segments:

$$\begin{matrix} & A & B \\ \bullet_E & F & [G] \\ & L & M \end{matrix}$$

$$f = A + L + \bullet_E$$
$$g = B + M$$

$$\begin{bmatrix} 4 & -1 \\ -2 & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix}$$

$$F' = \tfrac{1}{14}(4f + g)$$
$$G' = \tfrac{1}{14}(2f + 4g)$$

$$\phi = 4(1-\omega)F + \omega(A + L + \bullet_E)$$
$$\gamma = 4(1-\omega)G + \omega(B + M)$$

$$\begin{bmatrix} 4 & -\omega \\ -2\omega & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \end{bmatrix}$$

$$F' = \frac{4\phi + \omega\gamma}{16 - 2\omega^2}$$
$$G' = \frac{2\omega\phi + 4\gamma}{16 - 2\omega^2}$$

3-point row segments:

$$\begin{matrix} & A & B & C \\ \bullet_E & F & G & [H] \\ & L & M & N \end{matrix}$$

$$f = A + L + \bullet_E$$
$$g = B + M$$
$$h = C + N$$

$$\phi = 4(1-\omega)F + \omega(A + L + \bullet_E)$$
$$\gamma = 4(1-\omega)G + \omega(B + M)$$
$$\eta = 4(1-\omega)H + \omega(C + N)$$

$$\begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -2 & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega & 0 \\ -\omega & 4 & -\omega \\ 0 & -2\omega & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \end{bmatrix}$$

$$F' = \tfrac{1}{52}(14f + 4g + 1h) \qquad F' = \frac{(16 - 2\omega^2)\phi + 4\omega\gamma + \omega^2\eta}{4(16 - 3\omega^2)}$$

$$G' = \tfrac{1}{52}(4f + 16g + 4h) \qquad G' = \frac{4\omega\phi + 16\gamma + 4\omega\eta}{4(16 - 3\omega^2)}$$

$$H' = \tfrac{1}{52}(2f + 8g + 15h) \qquad H' = \frac{2\omega^2\phi + 8\omega\gamma + (16 - \omega^2)\eta}{4(16 - 3\omega^2)}$$

4-point row segments:

$$\begin{array}{cccc} & A & B & C & D \\ \bullet_E & F & G & H & [J] \\ & L & M & N & P \end{array}$$

$$f = A + L + \bullet_E \qquad \phi = 4(1-\omega)F + \omega(A + L + \bullet_E)$$

$$g = B + M \qquad \gamma = 4(1-\omega)G + \omega(B + M)$$

$$h = C + N \qquad \eta = 4(1-\omega)H + \omega(C + N)$$

$$j = D + P \qquad \lambda = 4(1-\omega)J + \omega(D + P)$$

$$\begin{bmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -2 & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \\ j \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega & 0 & 0 \\ -\omega & 4 & -\omega & 0 \\ 0 & -\omega & 4 & -\omega \\ 0 & 0 & -2\omega & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \\ \lambda \end{bmatrix}$$

$$F' = \tfrac{1}{194}(52f + 14g + 4h + 1j)$$

$$G' = \tfrac{1}{194}(14f + 56g + 16h + 4j)$$

$$H' = \tfrac{1}{194}(4f + 16g + 60h + 15j)$$

$$J' = \tfrac{1}{194}(2f + 8g + 30h + 56j)$$

[see D.4.]

D.2.6 Pure Neumann Boundaries

The situation of Neumann boundaries imposed on both endpoints of a line segment can only occur when the line segment is an *entire inner row* of a parallel plate model. Since there are many points on a line, generally more than 100 points, precalculated inverses are not utilized in practice and are therefore not provided.

It can be shown that the generalized matrix M of coefficients of the 5-point operator equations has the form

$$M = \begin{bmatrix} 4 & -2 & 0 & 0 & 0 \\ -1 & 4 & \ddots & 0 & 0 \\ 0 & \ddots & \ddots & -1 & 0 \\ 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & -2 & 4 \end{bmatrix}, \tag{D-4}$$

which can be easily inverted by tridiagonal algorithms [PTFV92].

D.3. Vertical Lines

The solution of vertical lines, or columns, of points is generally performed before or after the solution of all rows of points to enhance the convergence rate.

D.3.1 Typical Columns of Points

Parallel plate capacitors are modeled as having Dirichlet boundaries along the plates, and Neumann boundaries along the leftmost and rightmost columns. Figure D-3 shows several vertical lines of points from a discrete, mesh-based representation of a parallel plate capacitor. The leftmost highlighted column has Dirichlet boundaries at it endpoints and Neumann boundary conditions imposed on *all its points* due to the left side Neumann boundary of the capacitor. The rightmost highlighted column has Dirichlet

boundaries at it endpoints and Neumann boundary conditions imposed on *all its points* due to the right side Neumann boundary of the capacitor. The middle highlighted column contains 2 enclosed singularities due to the corona branches "cutting" or crossing that column. The properly segmented column consists of 3 line segments, such that:

- the top segment contains a Dirichlet boundary due to the known upper plate potential, and a Dirichlet boundary due to the uppermost singularity,
- the middle segment has just Dirichlet boundaries due to the singularities at its endpoints, and
- the bottom segment contains a Dirichlet boundary due to the known lower plate potential, and a Dirichlet boundary due to the lower singularity.

Thus, for columns of points, we have 3 possible combinations of boundaries imposed upon any segment. During the solution of any given column, *both adjacent columns* are treated as temporary Dirichlet boundaries.

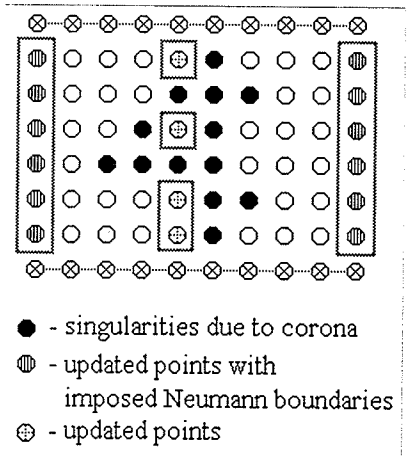


Fig. D - 3: Typical columns of points.

It can be shown that the generalized matrix M of coefficients of the 5-point operator equations for <u>any</u> *vertical (column) segment* has the form

$$M = \begin{bmatrix} 4 & -1 & 0 & 0 & 0 \\ -1 & 4 & \ddots & 0 & 0 \\ 0 & \ddots & \ddots & -1 & 0 \\ 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & -1 & 4 \end{bmatrix}. \qquad \text{(D-5)}$$

D.3.2 Pure Dirichlet Boundaries

D.3.2.1 Direct Solution of Short Segments

1-point column segments:

$$\begin{array}{ccc} & \bullet_E & \\ L & F & A \\ & \bullet_K & \end{array}$$

$$F' = \tfrac{1}{4}(A + L + \bullet_E + \bullet_K) \qquad F' = (1-\omega)F + \frac{\omega}{4}(A + L + \bullet_E + \bullet_K)$$

2-point column segments:

$$\begin{array}{ccc} & \bullet_E & \\ L & F & A \\ M & G & B \\ & \bullet_K & \end{array}$$

$$f = A + L + \bullet_E \qquad \phi = 4(1-\omega)F + \omega(A + L + \bullet_E)$$
$$g = B + M + \bullet_K \qquad \gamma = 4(1-\omega)G + \omega(B + M + \bullet_K)$$

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega \\ -\omega & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \end{bmatrix}$$

$$F' = \tfrac{1}{15}(4f + g) \qquad F' = \frac{4\phi + \omega\gamma}{16 - \omega^2}$$
$$G' = \tfrac{1}{15}(f + 4g) \qquad G' = \frac{\omega\phi + 4\gamma}{16 - \omega^2}$$

3-point column segments:

$\bullet_E$

$L \quad F \quad A$

$M \quad G \quad B$

$N \quad H \quad C$

$\bullet_K$

$$f = A + L + \bullet_E$$
$$g = B + M$$
$$h = C + N + \bullet_K$$

$$\begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \end{bmatrix}$$

$$F' = \tfrac{1}{56}(15f + 4g + 1h)$$
$$G' = \tfrac{1}{56}(4f + 16g + 4h)$$
$$H' = \tfrac{1}{56}(1f + 4g + 15h)$$

$$\phi = 4(1-\omega)F + \omega(A + L + \bullet_E)$$
$$\gamma = 4(1-\omega)G + \omega(B + M)$$
$$\eta = 4(1-\omega)H + \omega(C + N + \bullet_K)$$

$$\begin{bmatrix} 4 & -\omega & 0 \\ -\omega & 4 & -\omega \\ 0 & -\omega & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \end{bmatrix}$$

$$F' = \frac{(16-\omega^2)\phi + 4\omega\gamma + \omega^2\eta}{8(8-\omega^2)}$$
$$G' = \frac{4\omega\phi + 16\gamma + 4\omega\eta}{8(8-\omega^2)}$$
$$H' = \frac{\omega^2\phi + 4\omega\gamma + (16-\omega^2)\eta}{8(8-\omega^2)}$$

4-point column segments:

$\bullet_E$

$L \quad F \quad A$

$M \quad G \quad B$

$N \quad H \quad C$

$P \quad J \quad D$

$\bullet_K$

$$f = A + L + \bullet_E \qquad \phi = 4(1-\omega)F + \omega(A + L + \bullet_E)$$
$$g = B + M \qquad \gamma = 4(1-\omega)G + \omega(B + M)$$
$$h = C + N \qquad \eta = 4(1-\omega)H + \omega(C + N)$$
$$j = D + P + \bullet_K \qquad \lambda = 4(1-\omega)J + \omega(D + P + \bullet_K)$$

$$\begin{bmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \\ j \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega & 0 & 0 \\ -\omega & 4 & -\omega & 0 \\ 0 & -\omega & 4 & -\omega \\ 0 & 0 & -\omega & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \\ \lambda \end{bmatrix}$$

$$F' = \tfrac{1}{209}(56f + 15g + 4h + 1j)$$
$$G' = \tfrac{1}{209}(15f + 60g + 16h + 4j)$$
$$H' = \tfrac{1}{209}(4f + 16g + 60h + 15j)$$
$$J' = \tfrac{1}{209}(1f + 4g + 15h + 56j)$$

[see D.4.]

D.3.3 Left Neumann Boundaries

All points on column segments from the leftmost column of mesh points will experience Neumann boundary conditions. This situation occurs when solving the potentials along the leftmost column of the mesh used for modelling a parallel plate capacitor.

D.3.3.1 Direct Solution of Short Segments

1-point column segments:

$\bullet_E$
$[F] \quad A$
$\bullet_K$

$$F' = \tfrac{1}{4}(2A + \bullet_E + \bullet_K) \qquad F' = (1-\omega)F + \frac{\omega}{4}(2A + \bullet_E + \bullet_K)$$

2-point column segments:

$\bullet_E$

$[F]$ A

$[G]$ B

$\bullet_K$

$$f = 2A + \bullet_E \qquad \phi = 4(1-\omega)F + \omega(2A + \bullet_E)$$
$$g = 2B + \bullet_K \qquad \gamma = 4(1-\omega)G + \omega(2B + \bullet_K)$$

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega \\ -\omega & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \end{bmatrix}$$

$$F' = \tfrac{1}{15}(4f + g) \qquad F' = \frac{4\phi + \omega\gamma}{16 - \omega^2}$$
$$G' = \tfrac{1}{15}(f + 4g) \qquad G' = \frac{\omega\phi + 4\gamma}{16 - \omega^2}$$

3-point column segments:

$\bullet_E$

$[F]$ A

$[G]$ B

$[H]$ C

$\bullet_K$

$$f = 2A + \bullet_E \qquad \phi = 4(1-\omega)F + \omega(2A + \bullet_E)$$
$$g = 2B \qquad \gamma = 4(1-\omega)G + \omega(2B)$$
$$h = 2C + \bullet_K \qquad \eta = 4(1-\omega)H + \omega(2C + \bullet_K)$$

$$\begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega & 0 \\ -\omega & 4 & -\omega \\ 0 & -\omega & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \end{bmatrix}$$

$$F' = \tfrac{1}{56}(15f + 4g + 1h)$$
$$G' = \tfrac{1}{56}(4f + 16g + 4h)$$
$$H' = \tfrac{1}{56}(1f + 4g + 15h)$$

$$F' = \frac{(16-\omega^2)\phi + 4\omega\gamma + \omega^2\eta}{8(8-\omega^2)}$$
$$G' = \frac{4\omega\phi + 16\gamma + 4\omega\eta}{8(8-\omega^2)}$$
$$H' = \frac{\omega^2\phi + 4\omega\gamma + (16-\omega^2)\eta}{8(8-\omega^2)}$$

4-point column segments:

$\bullet_E$
$[F]$ A
$[G]$ B
$[H]$ C
$[J]$ D
$\bullet_K$

$$f = 2A + \bullet_E$$
$$g = 2B$$
$$h = 2C$$
$$j = 2D + \bullet_K$$

$$\phi = 4(1-\omega)F + \omega(2A + \bullet_E)$$
$$\gamma = 4(1-\omega)G + \omega(2B)$$
$$\eta = 4(1-\omega)H + \omega(2C)$$
$$\lambda = 4(1-\omega)J + \omega(2D + \bullet_K)$$

$$\begin{bmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \\ j \end{bmatrix}$$

$$\begin{bmatrix} 4 & -\omega & 0 & 0 \\ -\omega & 4 & -\omega & 0 \\ 0 & -\omega & 4 & -\omega \\ 0 & 0 & -\omega & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \\ \lambda \end{bmatrix}$$

$$F' = \tfrac{1}{209}(56f + 15g + 4h + 1j)$$
$$G' = \tfrac{1}{209}(15f + 60g + 16h + 4j)$$
$$H' = \tfrac{1}{209}(4f + 16g + 60h + 15j)$$
$$J' = \tfrac{1}{209}(1f + 4g + 15h + 56j)$$

[see D.4.]

D.3.4 Right Neumann Boundaries

All points on column segments from the rightmost column of mesh points will experience Neumann boundary conditions. This situation occurs when solving the

potentials along the rightmost column of the mesh used for modelling a parallel plate capacitor.

D.3.4.1 Direct Solution of Short Segments

1-point column segments:

$$\begin{matrix} & \bullet_E \\ L & [F] \\ & \bullet_K \end{matrix}$$

$$F' = \tfrac{1}{4}(2L + \bullet_E + \bullet_K) \qquad F' = (1-\omega)F + \frac{\omega}{4}(2L + \bullet_E + \bullet_K)$$

2-point column segments:

$$\begin{matrix} & \bullet_E \\ L & [F] \\ M & [G] \\ & \bullet_K \end{matrix}$$

$$f = 2L + \bullet_E \qquad \phi = 4(1-\omega)F + \omega(2L + \bullet_E)$$
$$g = 2M + \bullet_K \qquad \gamma = 4(1-\omega)G + \omega(2M + \bullet_K)$$

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} f \\ g \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega \\ -\omega & 4 \end{bmatrix}\begin{bmatrix} F' \\ G' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \end{bmatrix}$$

$$F' = \tfrac{1}{15}(4f + g) \qquad F' = \frac{4\phi + \omega\gamma}{16 - \omega^2}$$
$$G' = \tfrac{1}{15}(f + 4g) \qquad G' = \frac{\omega\phi + 4\gamma}{16 - \omega^2}$$

3-point column segments:

$\bullet_E$

L $[F]$

M $[G]$

N $[H]$

$\bullet_K$

$$f = 2L + \bullet_E$$
$$g = 2M$$
$$h = 2N + \bullet_K$$

$$\begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \end{bmatrix}$$

$$F' = \tfrac{1}{56}(15f + 4g + 1h)$$
$$G' = \tfrac{1}{56}(4f + 16g + 4h)$$
$$H' = \tfrac{1}{56}(1f + 4g + 15h)$$

$$\phi = 4(1-\omega)F + \omega(2L + \bullet_E)$$
$$\gamma = 4(1-\omega)G + \omega(2M)$$
$$\eta = 4(1-\omega)H + \omega(2N + \bullet_K)$$

$$\begin{bmatrix} 4 & -\omega & 0 \\ -\omega & 4 & -\omega \\ 0 & -\omega & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \end{bmatrix}$$

$$F' = \frac{(16-\omega^2)\phi + 4\omega\gamma + \omega^2\eta}{8(8-\omega^2)}$$

$$G' = \frac{4\omega\phi + 16\gamma + 4\omega\eta}{8(8-\omega^2)}$$

$$H' = \frac{\omega^2\phi + 4\omega\gamma + (16-\omega^2)\eta}{8(8-\omega^2)}$$

4-point column segments:

$\bullet_E$

L $[F]$

M $[G]$

N $[H]$

P $[J]$

$\bullet_K$

$$f = 2L + \bullet_E \qquad \phi = 4(1-\omega)F + \omega(2L + \bullet_E)$$
$$g = 2M \qquad \gamma = 4(1-\omega)G + \omega(2M)$$
$$h = 2N \qquad \eta = 4(1-\omega)H + \omega(2N)$$
$$j = 2P + \bullet_K \qquad \lambda = 4(1-\omega)J + \omega(2P + \bullet_K)$$

$$\begin{bmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & -1 & 0 \\ 0 & -1 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} f \\ g \\ h \\ j \end{bmatrix} \qquad \begin{bmatrix} 4 & -\omega & 0 & 0 \\ -\omega & 4 & -\omega & 0 \\ 0 & -\omega & 4 & -\omega \\ 0 & 0 & -\omega & 4 \end{bmatrix} \begin{bmatrix} F' \\ G' \\ H' \\ J' \end{bmatrix} = \begin{bmatrix} \phi \\ \gamma \\ \eta \\ \lambda \end{bmatrix}$$

$$F' = 1/209\,(56f + 15g + 4h + 1j)$$
$$G' = 1/209\,(15f + 60g + 16h + 4j)$$
$$H' = 1/209\,(4f + 16g + 60h + 15j)$$
$$J' = 1/209\,(1f + 4g + 15h + 56j)$$

[see D.4.]

D.3.5 Pure Neumann Boundaries

Inherent to the model configurations examined in this thesis, no column segment can experience Neumann boundary conditions along its right *and* left sides simultaneously, for that would imply a non-sensical situation of a single column of points with open boundaries along either side.

D.4. Implementation

The tridiagonal inversion algorithm which is employed is simplified by having "1"s along the upper diagonal U and the lower diagonal L (see listings for *sline.cpp*). For non-accelerated methods, the diagonal D consists of -4's. For accelerated methods, U and L remained unchanged while elements of D consists of $-4/\omega$. Introduction of Neumann boundaries affects only the 1st element of U or the last element of L. For accelerated (SLOR) solution for segments containing more than 3 points, the tridiagonal inverse

algorithm is used instead of direct methods utilizing precalculated inverses. This is due to the added computational complexity when an acceleration factor is introduced.

The *methods.exe* program [see listings in Appendix G] was written in part to rigorously test and verify the code for the solution of Laplace's equation by the method of lines. The solutions for all possible short segments as well as some longer segments, for all possible boundary conditions, were tested and then verified using Mathcad®.

D.5. Summary

For our selected set of electrode configurations,

- row segments experience Dirichlet boundary conditions along adjacent rows and Neumann or Dirichlet boundary conditions at their endpoints, and
- column segments experience Dirichlet boundary conditions at their endpoints and Neumann or Dirichlet boundary conditions along their adjacent columns.

Experiments show that the introduction of

- Neumann boundaries does not increase the computational time since all cases are solved identically (from a methodology viewpoint), and
- singularities decreases the computational time due to the utilization of direct methods for solving short segments.

APPENDIX E

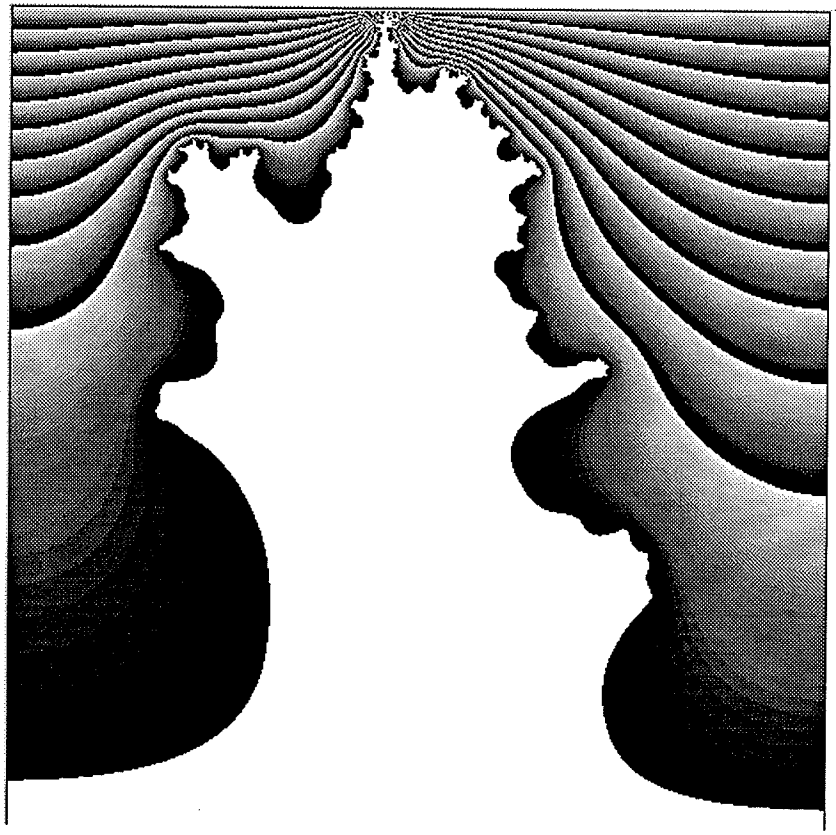
SIMULATED DIELECTRIC DISCHARGES

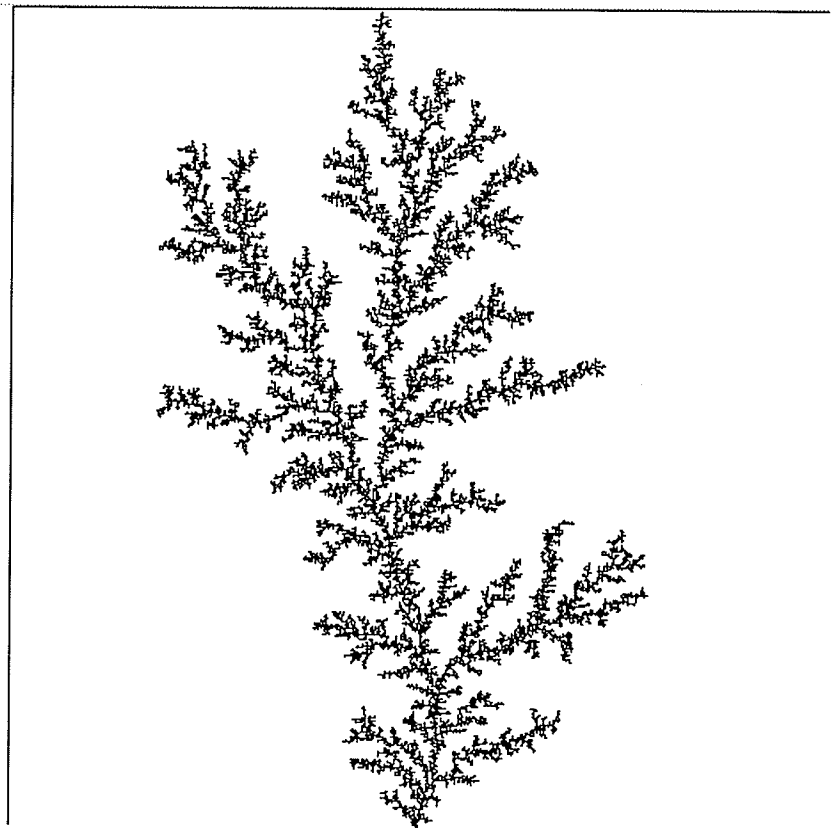
E.1. Introduction

Integral to the research for this thesis were simulations of dielectric discharges for varying parameters and methods. This appendix contains the resultant Laplacian fractals and their associated field distributions. The gray scale shaded field distributions were computed to verify that the iterative solution methods were "behaving" properly and that nothing was going awry in the global sense of the entire process. The parameters and conditions under which the Laplacian fractals were grown are discussed in Chapter 4.

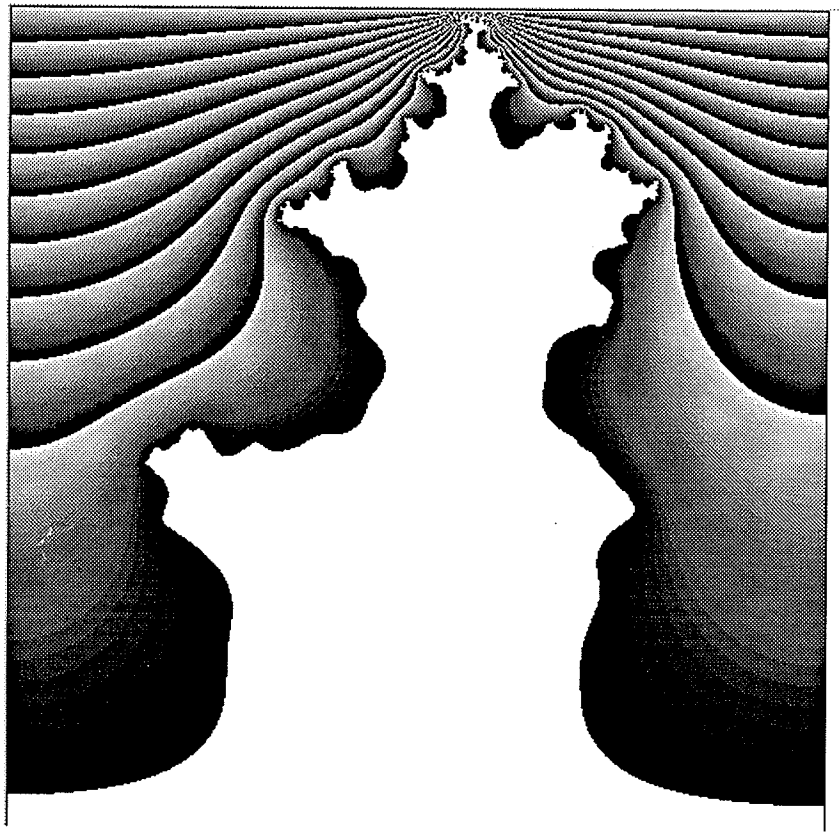
The software for the initial simulations failed to capture the lower and right sides of the frames containing the field patterns and the Laplacian fractals; I offer my sincere apologies for this initial programming mistake.

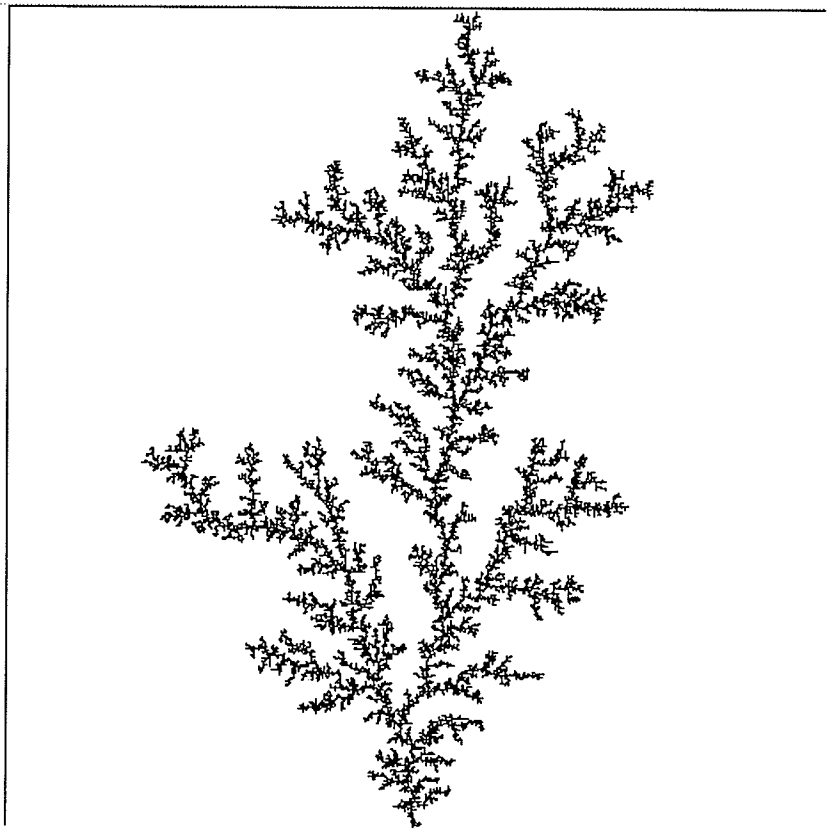
Filename	potentials	local iterations	global iterations	mesh scanning	tag scanning	random generator
trinn4	32-bit integer	GS	GS	normal	normal	rand2 (81)



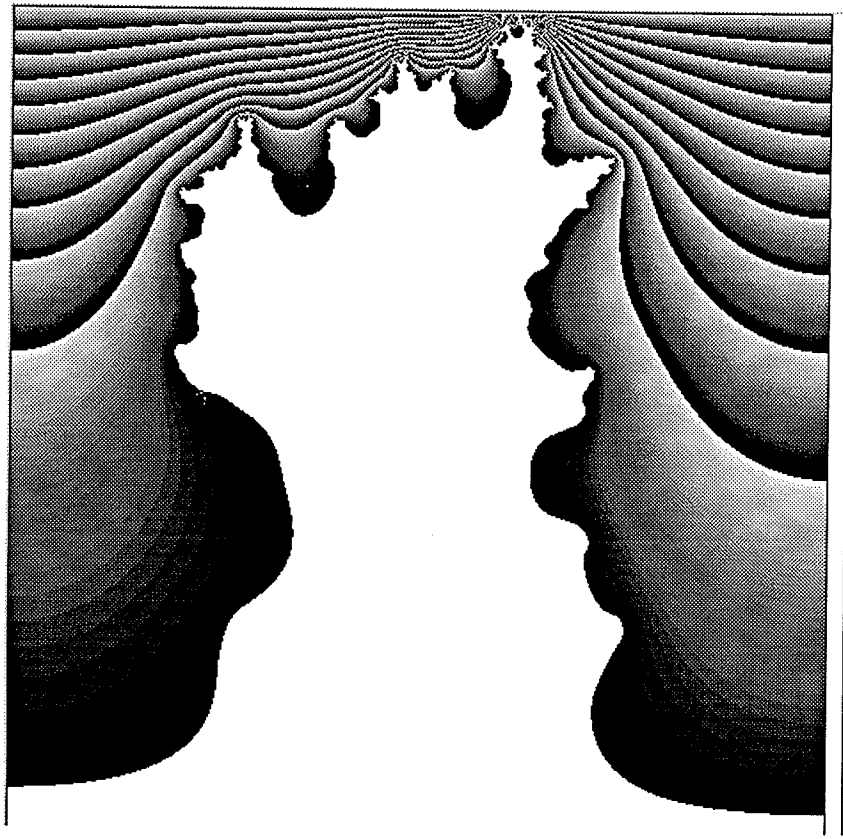


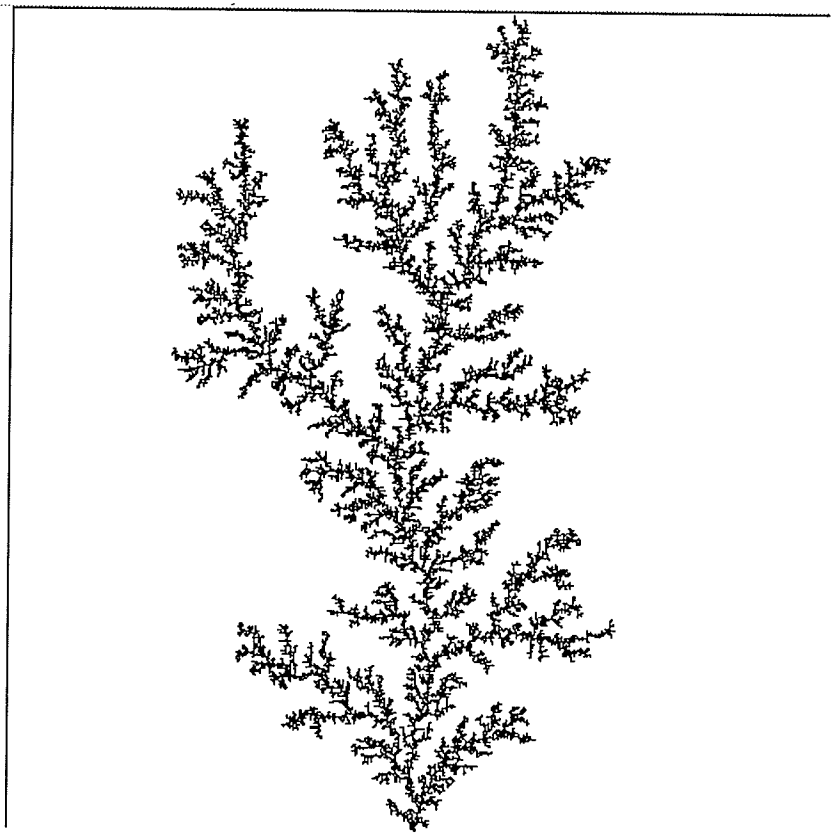
Filename	potentials	local iterations	global iterations	mesh scanning	tag scanning	random generator
trinr4	32-bit integer	GS	GS	normal	reverse	rand2 (81)



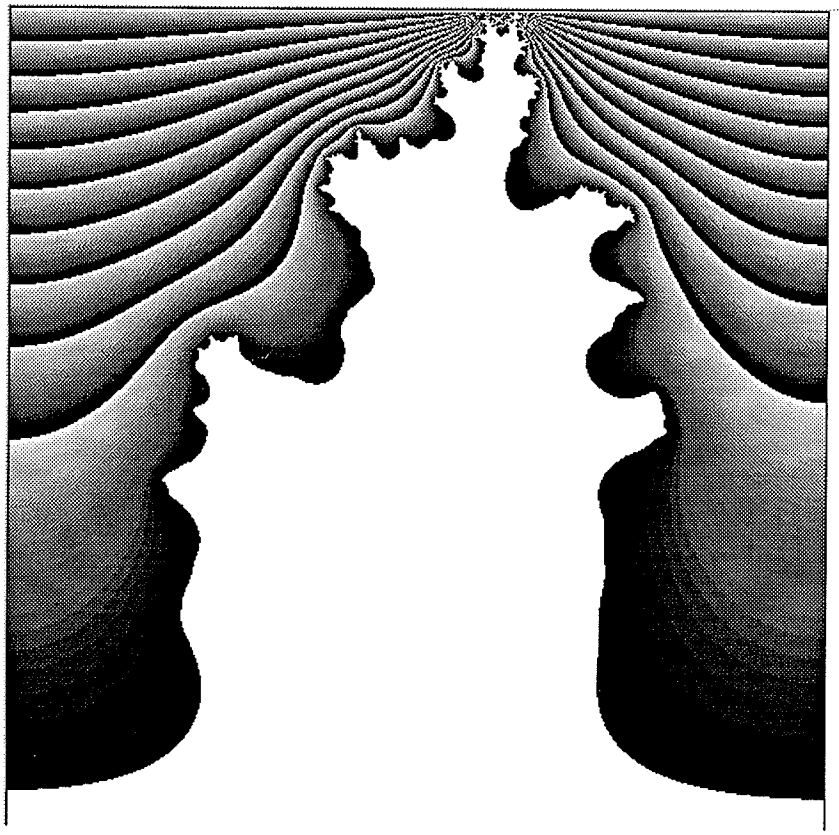


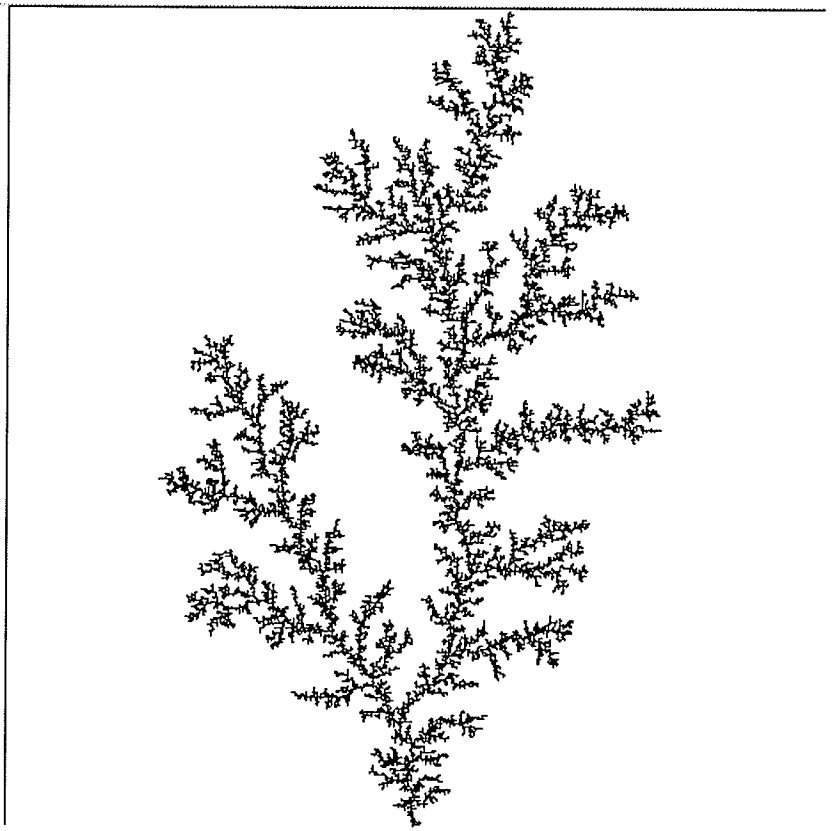
Filename	potentials	local iterations	global iterations	mesh scanning	tag scanning	random generator
trirn4	32-bit integer	GS	GS	reverse	normal	rand2 (81)



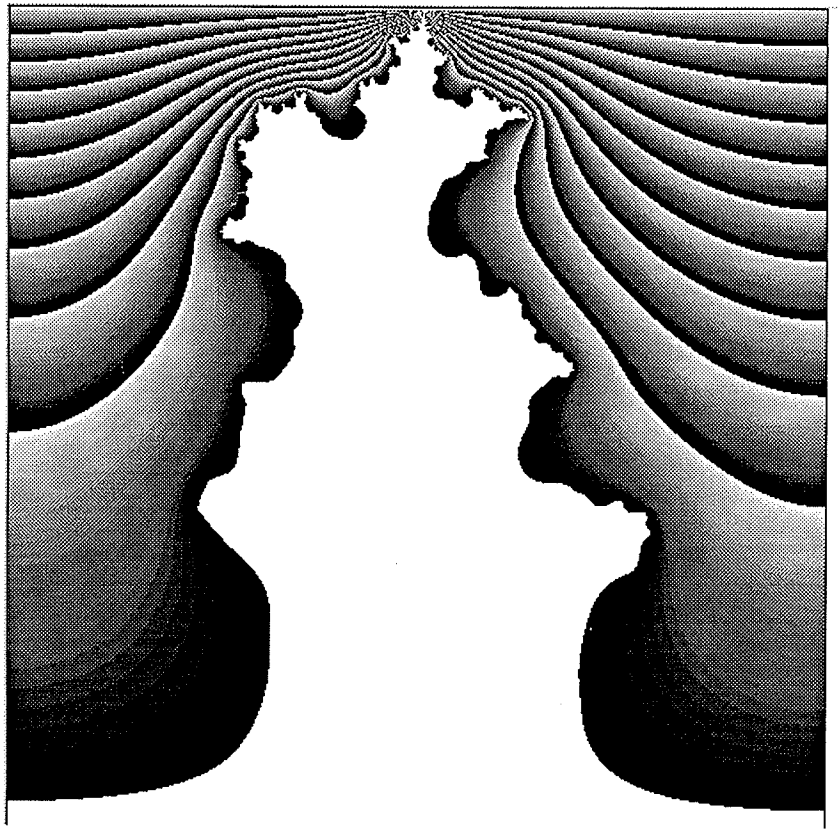


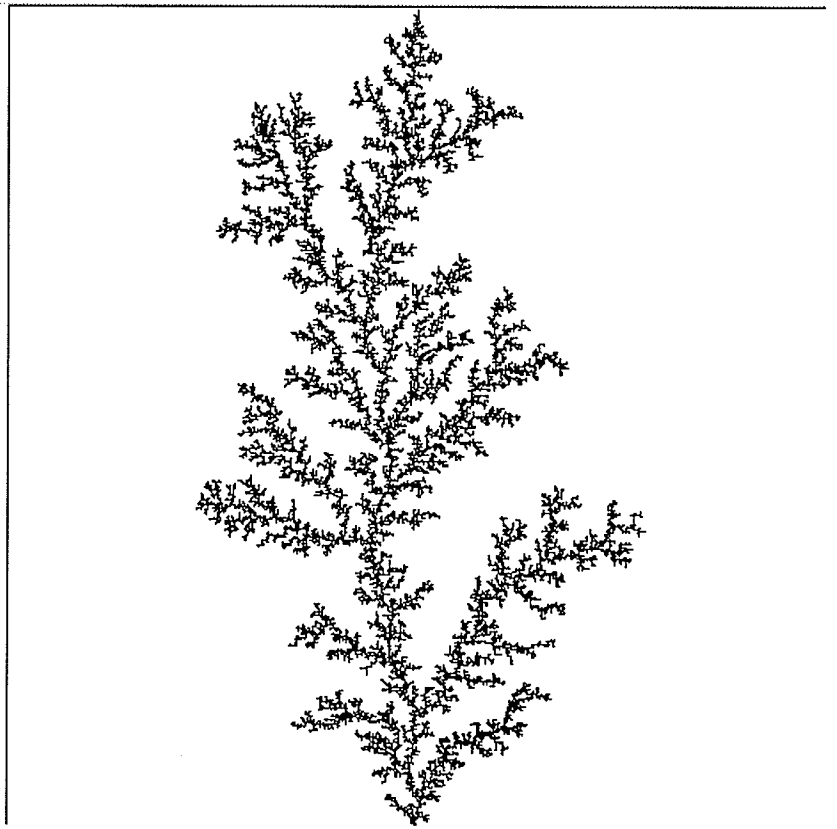
Filename	potentials	local iterations	global iterations	mesh scanning	tag scanning	random generator
trfnr4	32-bit floats	GS	GS	normal	reverse	rand2 (81)



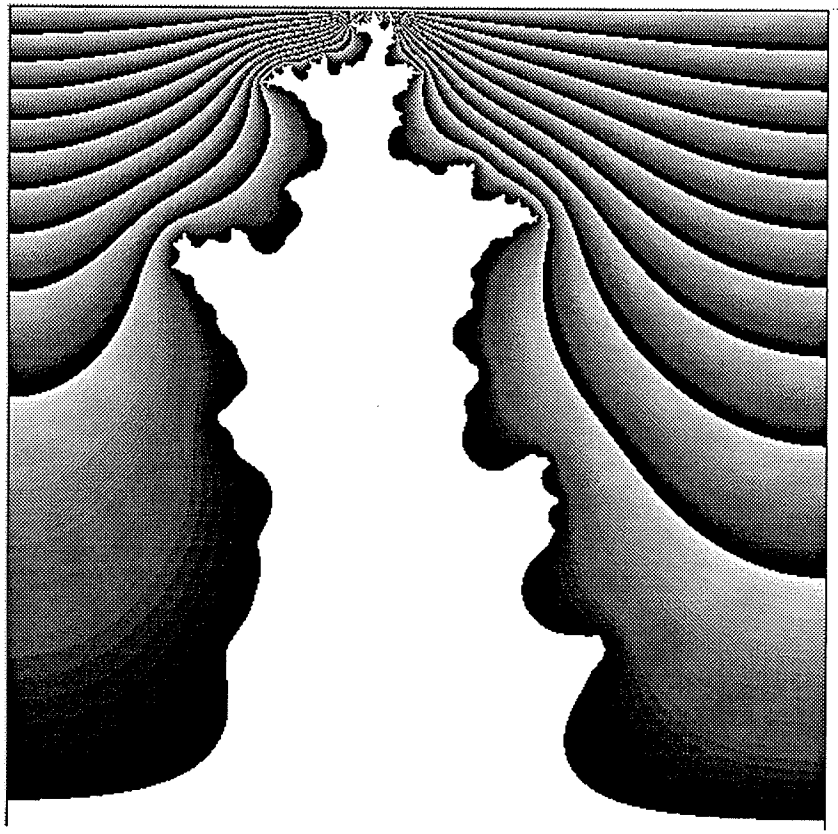


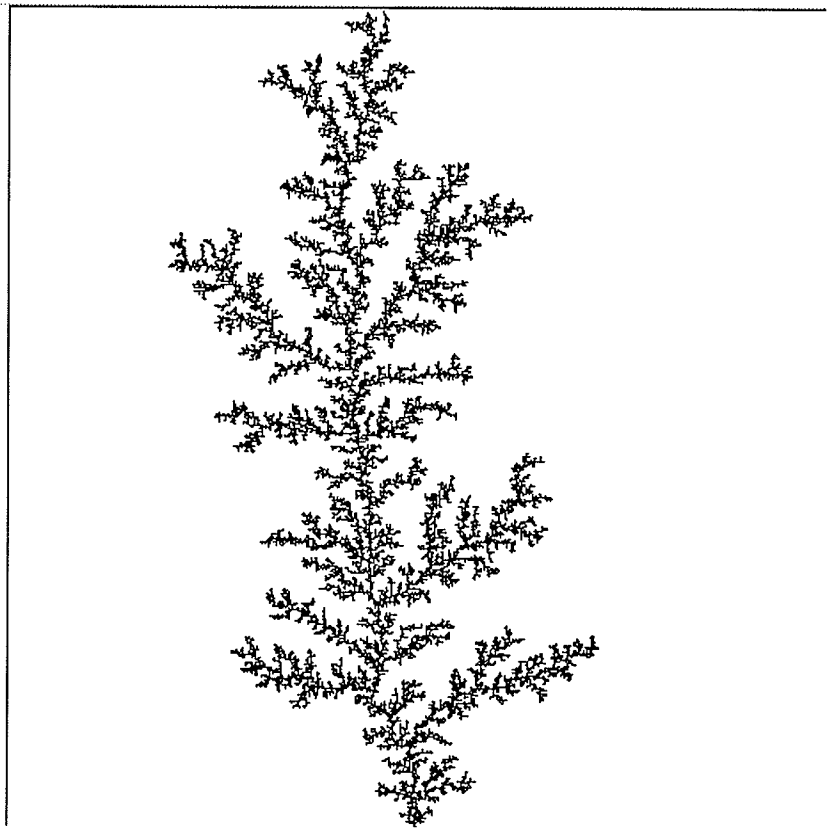
Filename	potentials	local iterations	global iterations	mesh scanning	tag scanning	random generator
trfrn4	32-bit floats	GS	GS	reverse	normal	rand2 (81)



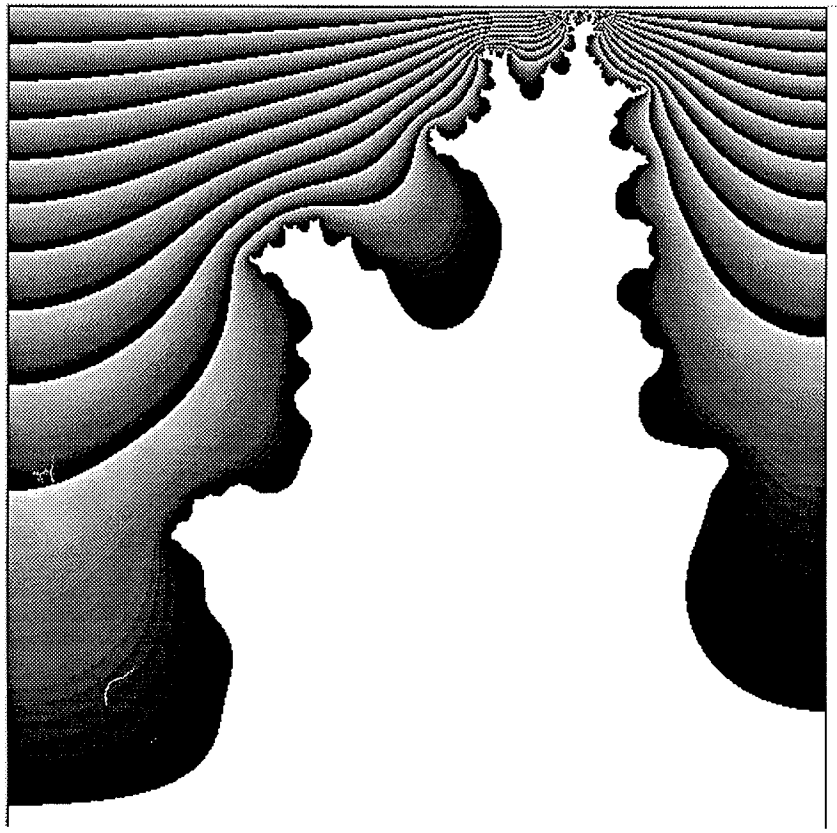


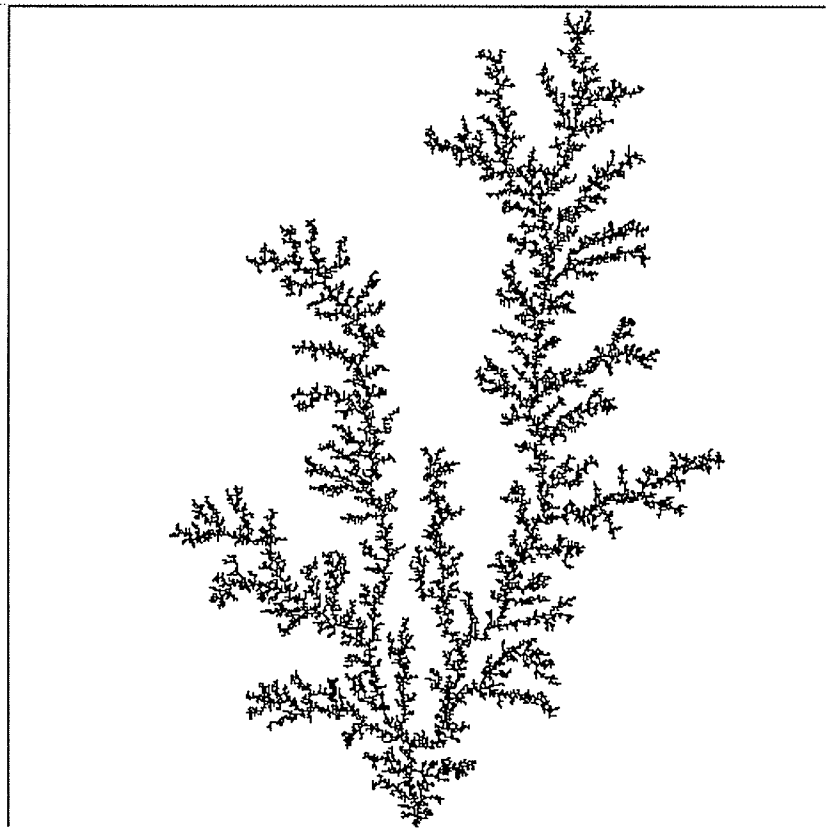
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn0	GS	GS	n/a	rand



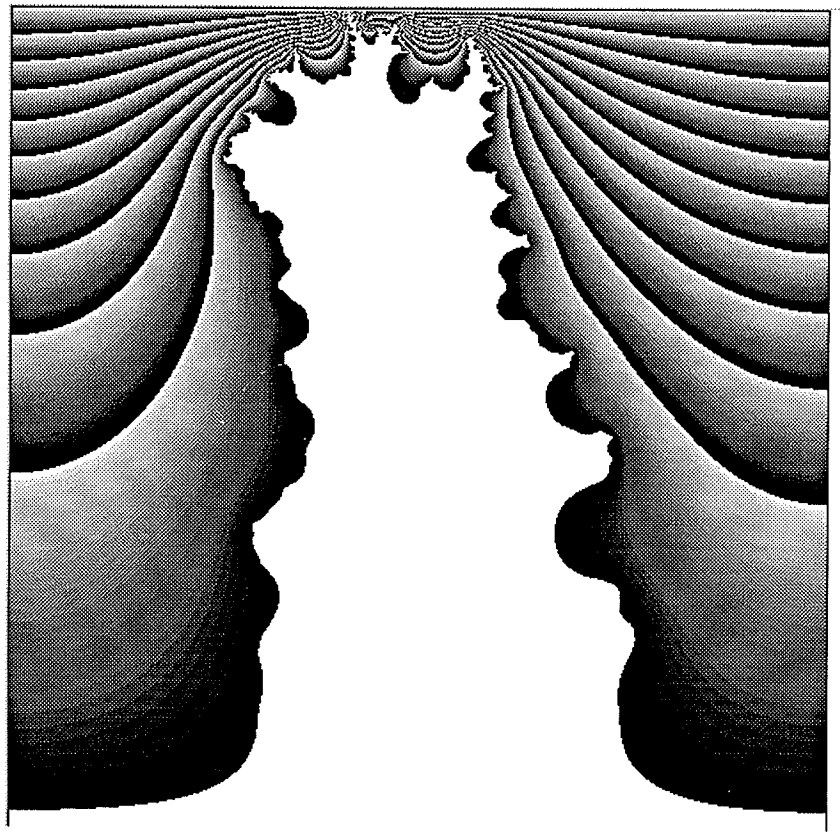


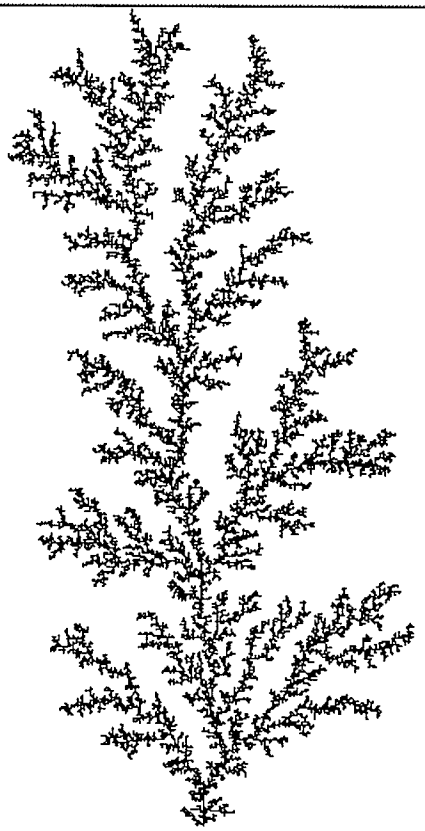
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn1	GS	GS	n/a	rand0



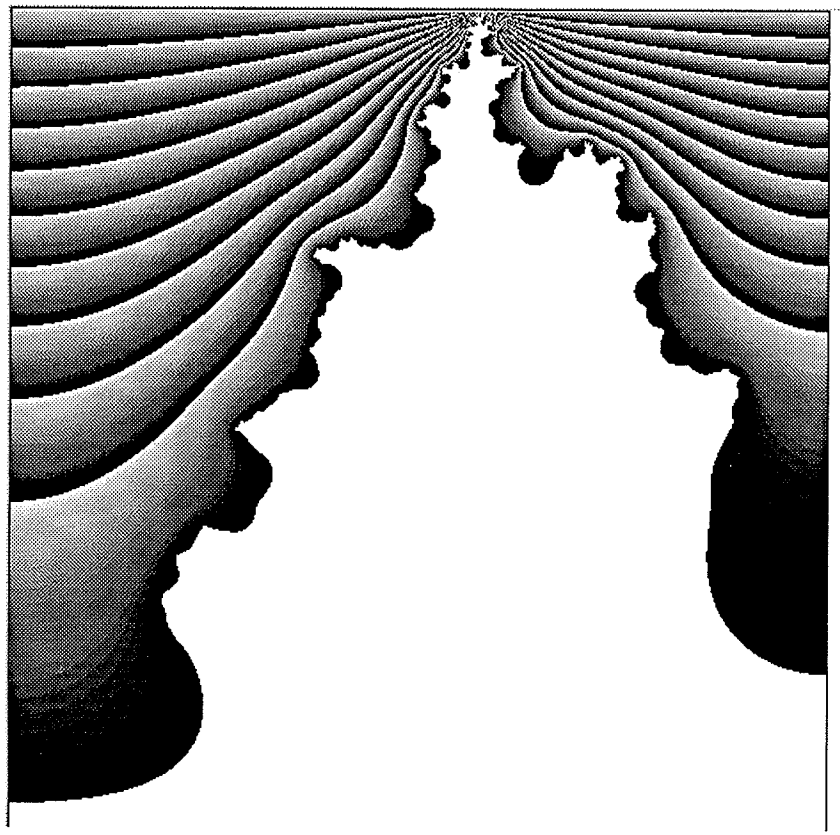


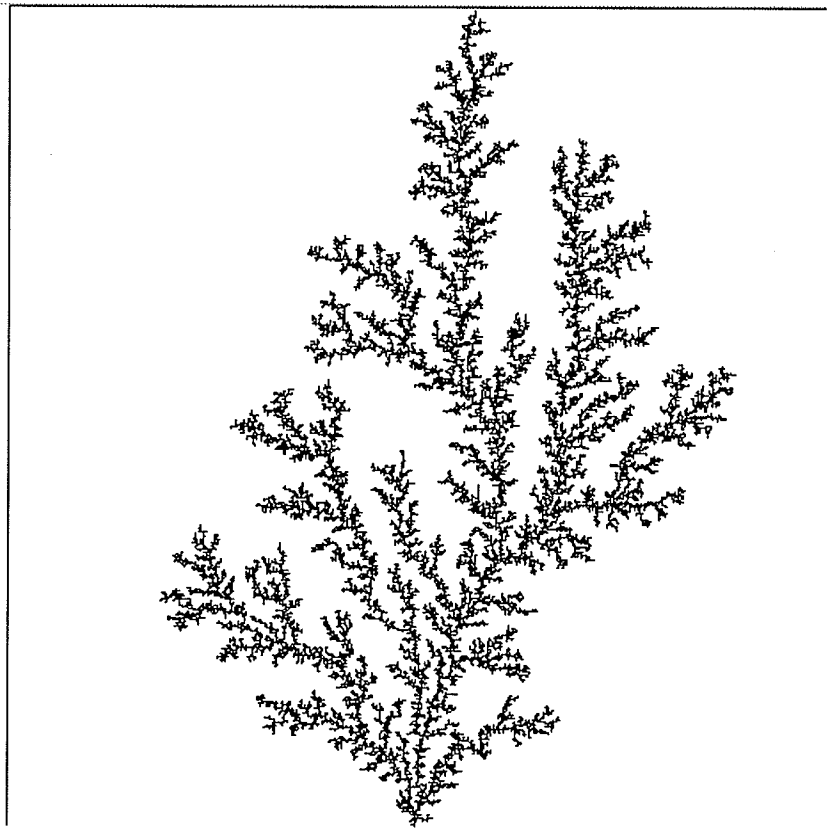
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn2	GS	GS	n/a	rand1



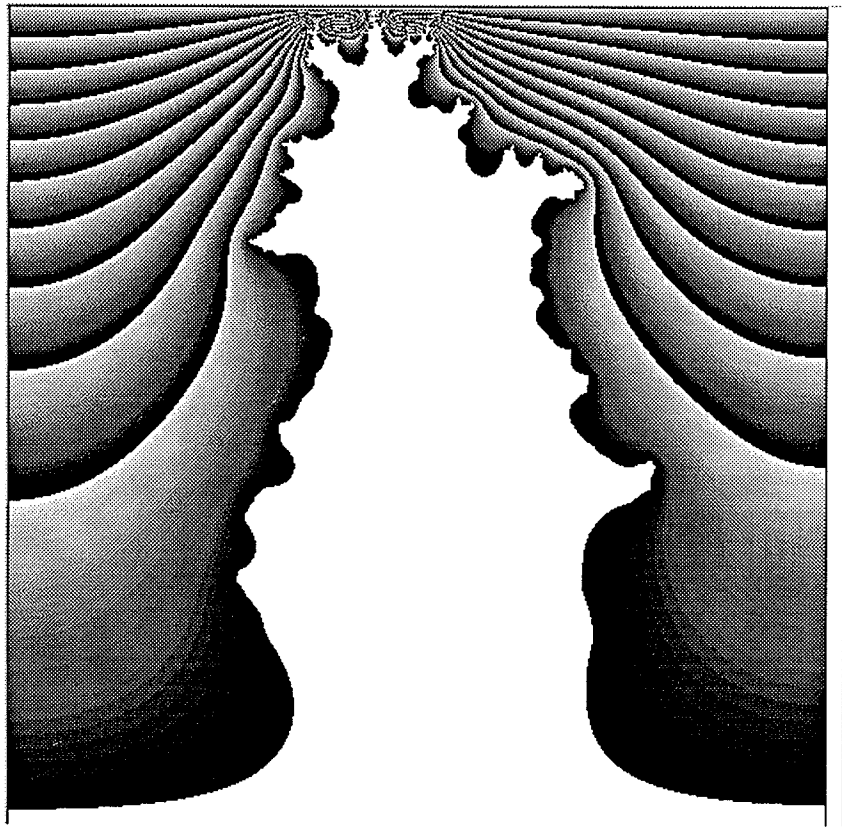


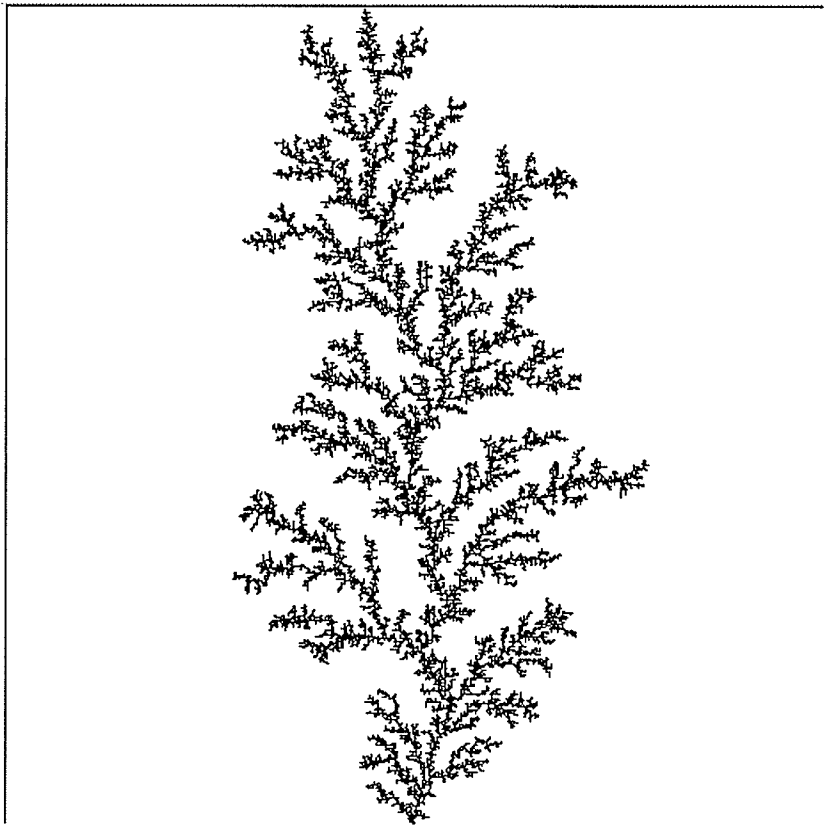
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn3	GS	GS	n/a	rand2





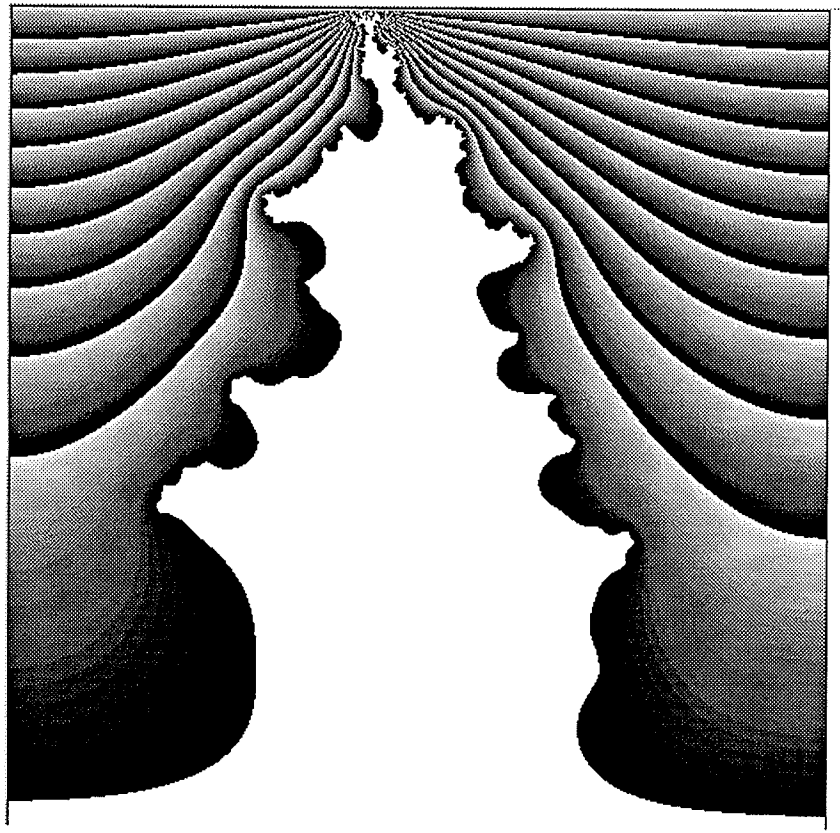
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn4	GS	GS	n/a	rand2 (81)

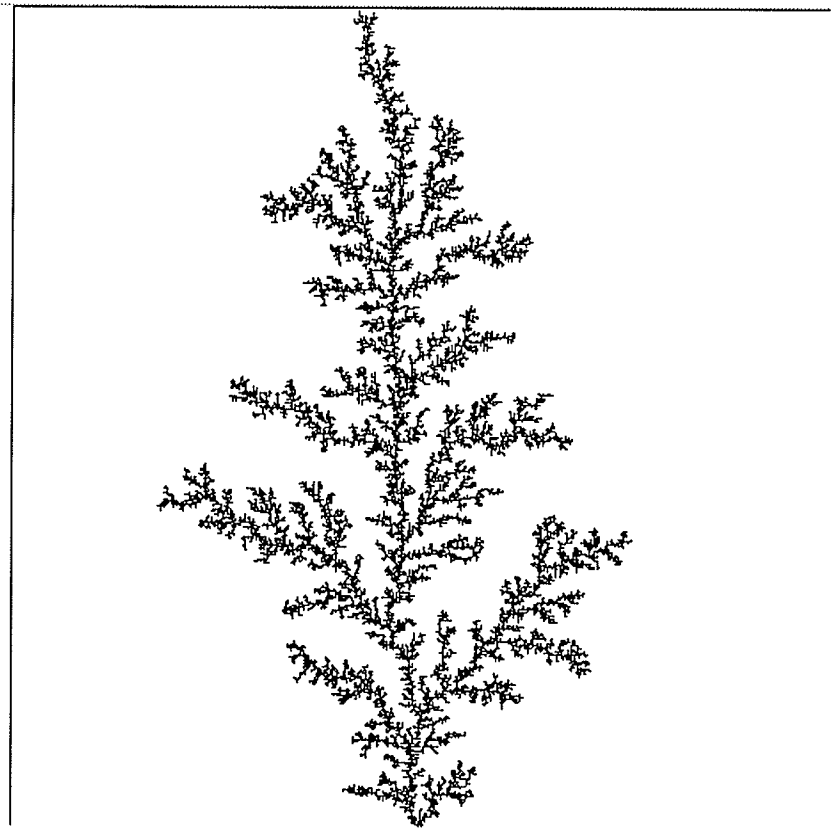




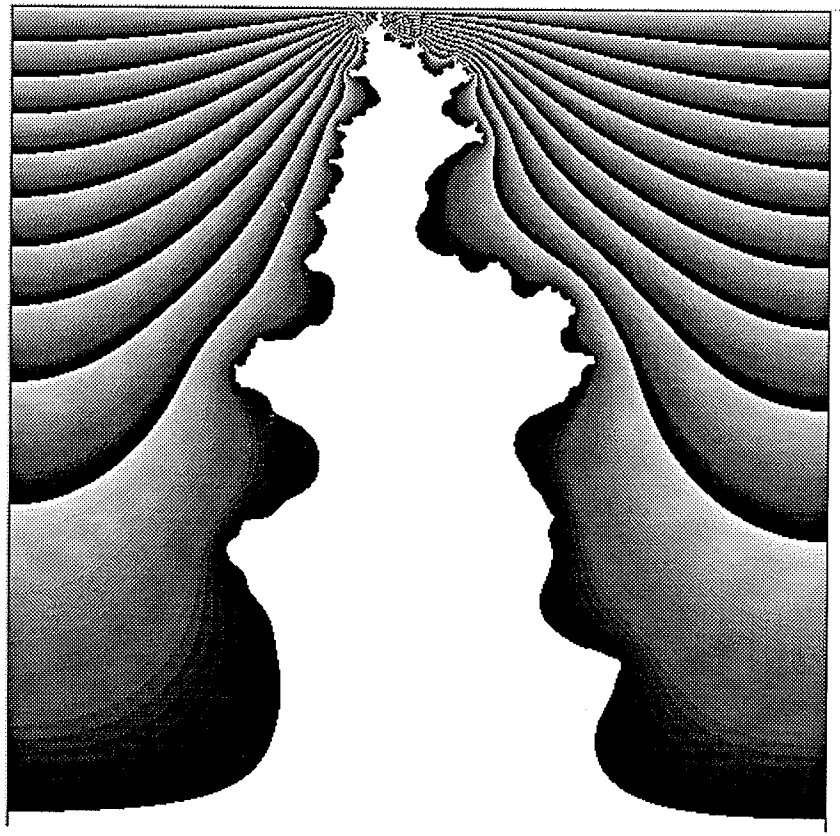
Note: used a 50×50 local update region. Compare with **app220**.

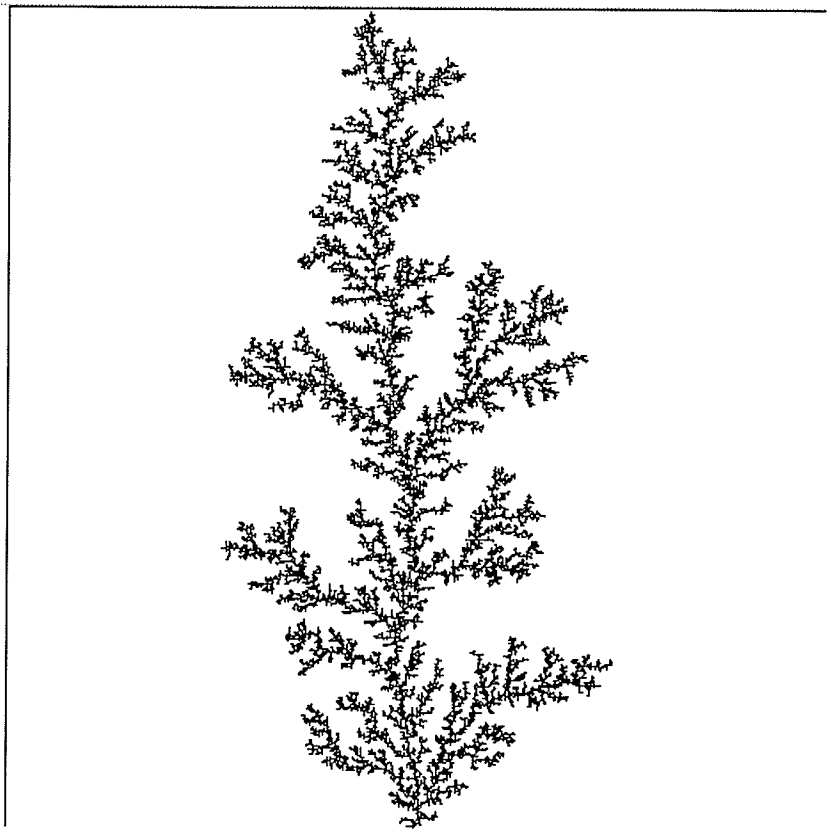
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn5	GS	GS	n/a	rand3



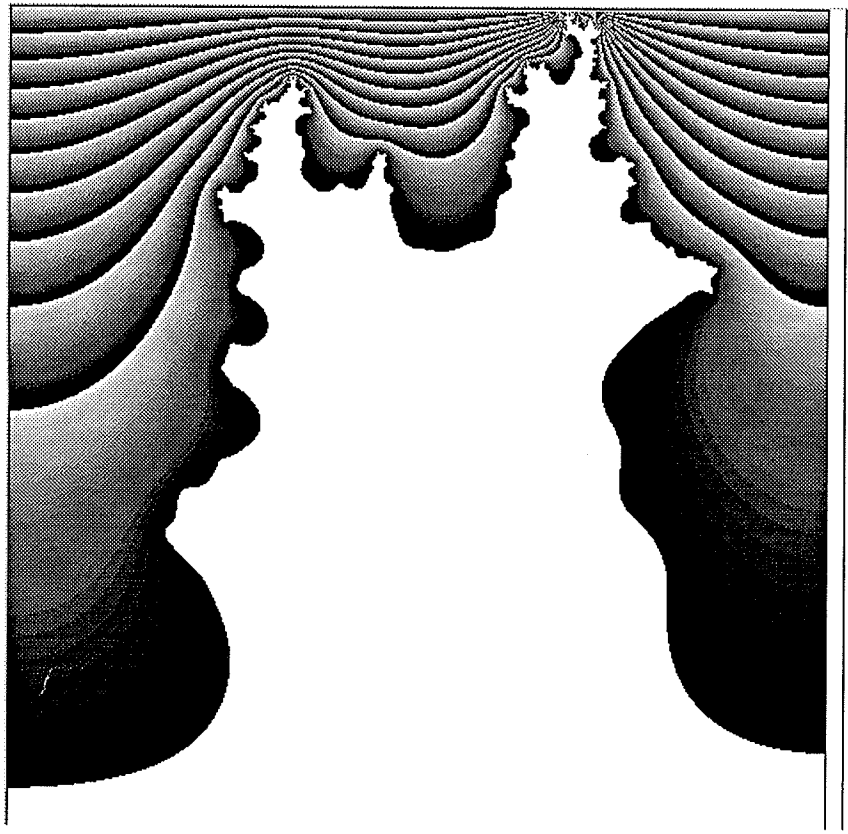


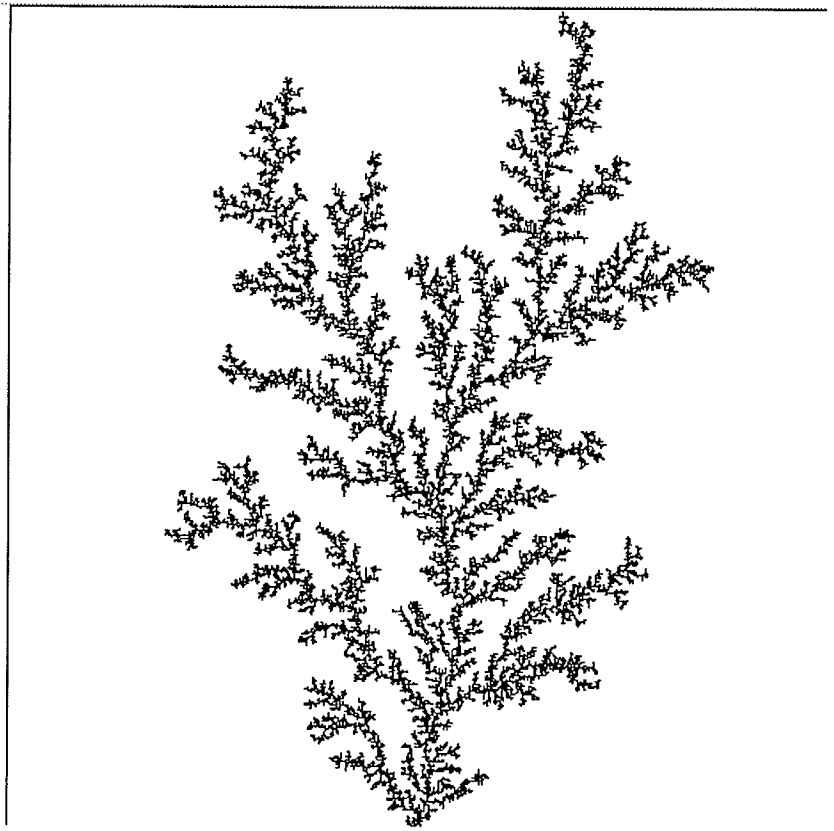
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn6	GS	GS	n/a	randq1



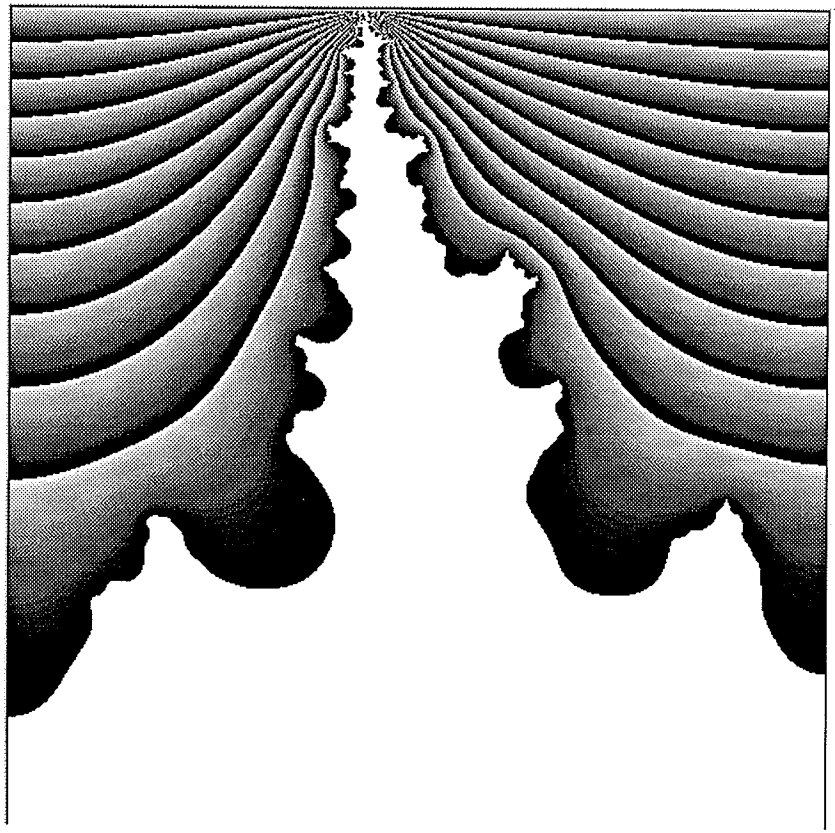


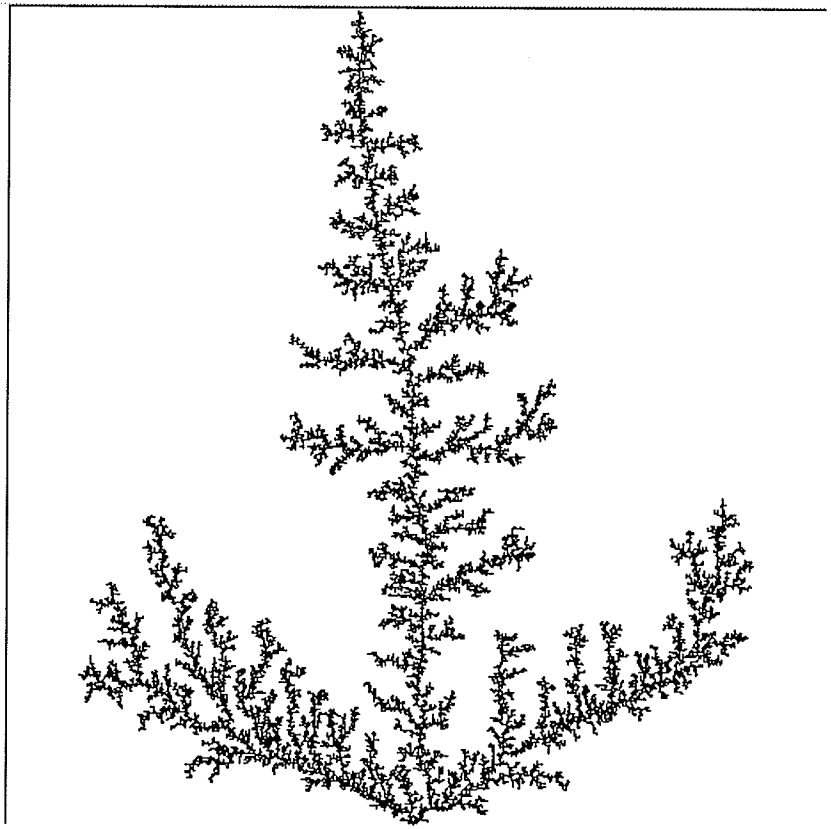
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn7	GS	GS	n/a	randq2



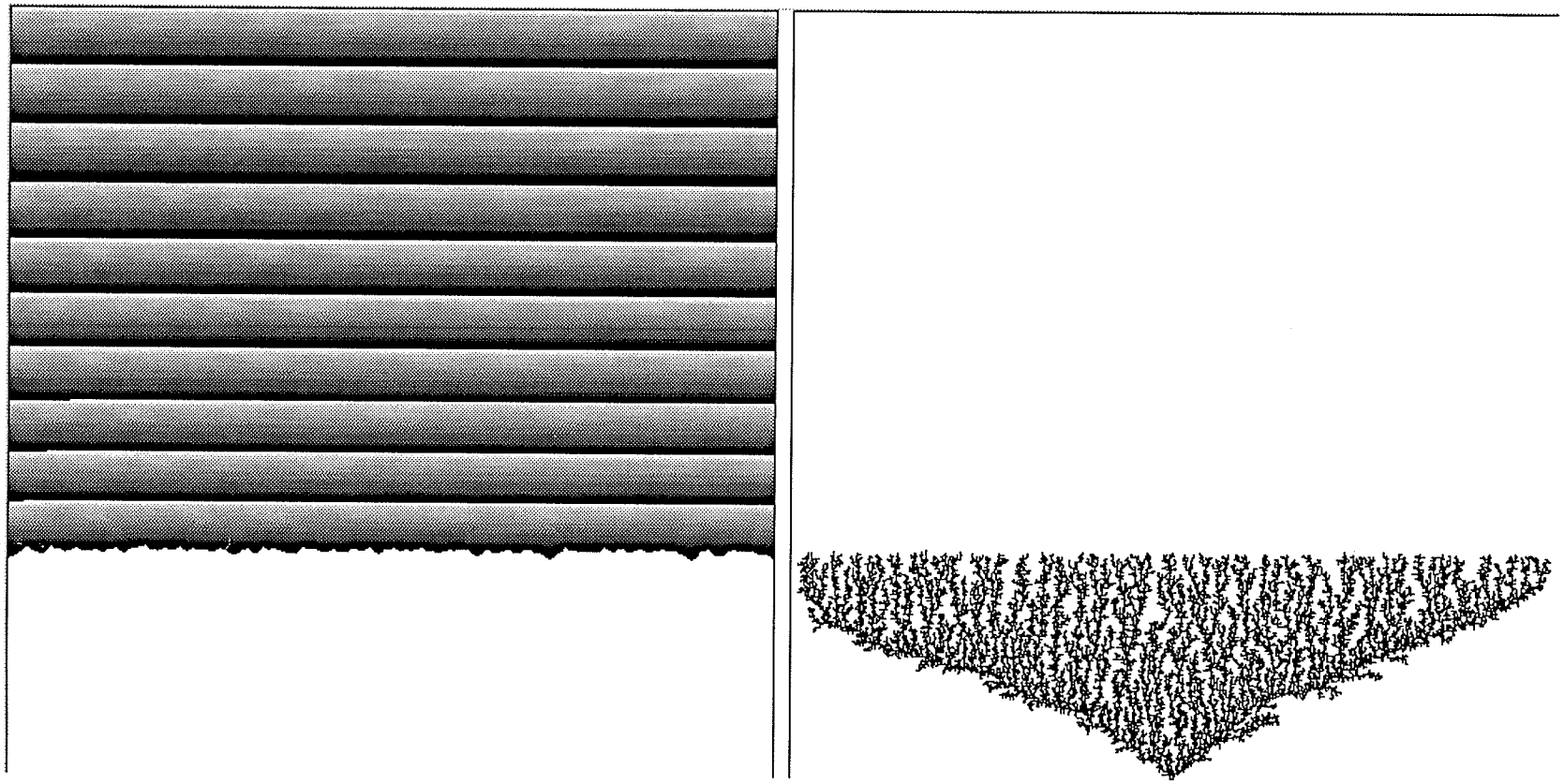


Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn8	GS	GS	n/a	Hénon

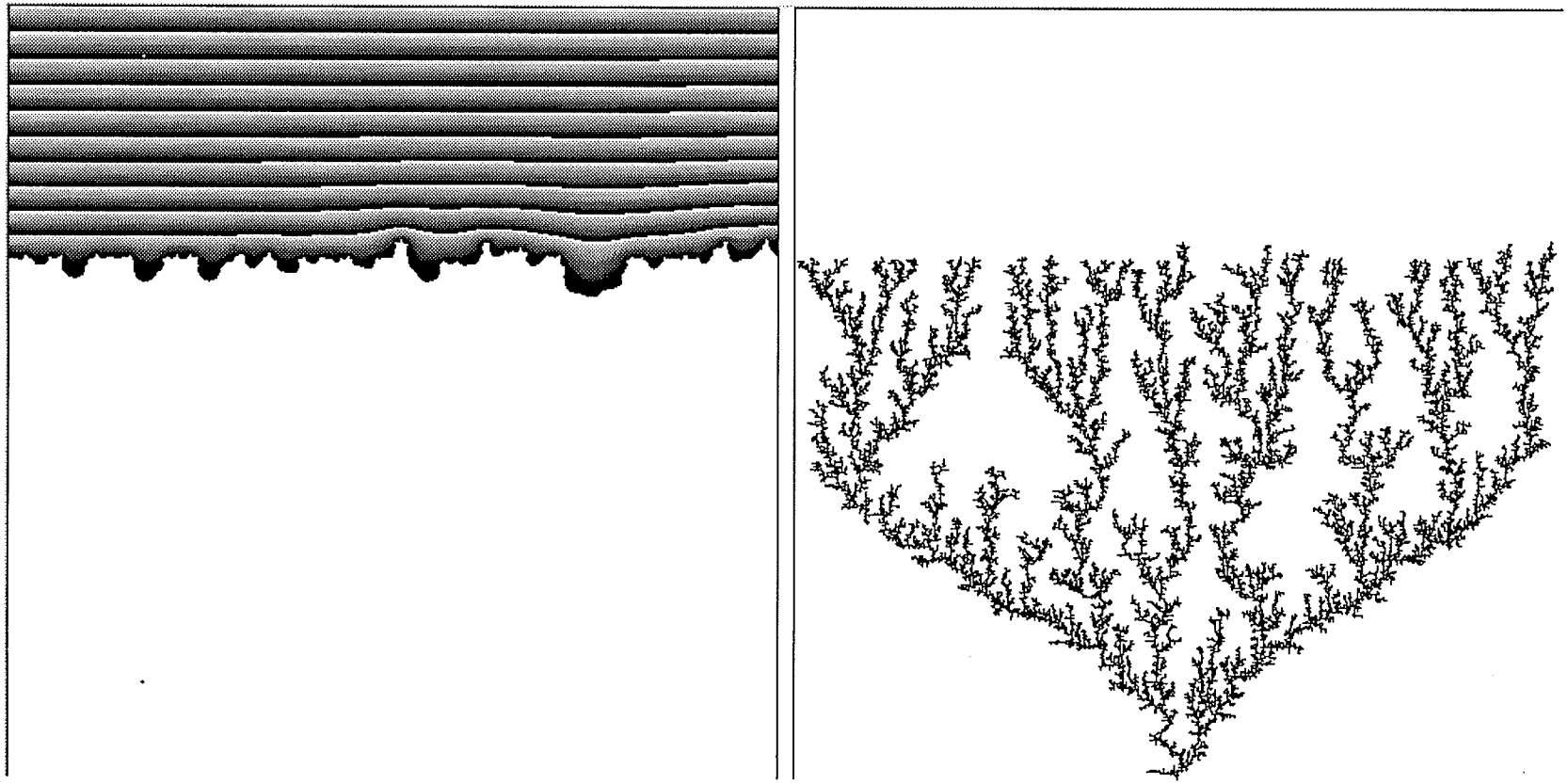




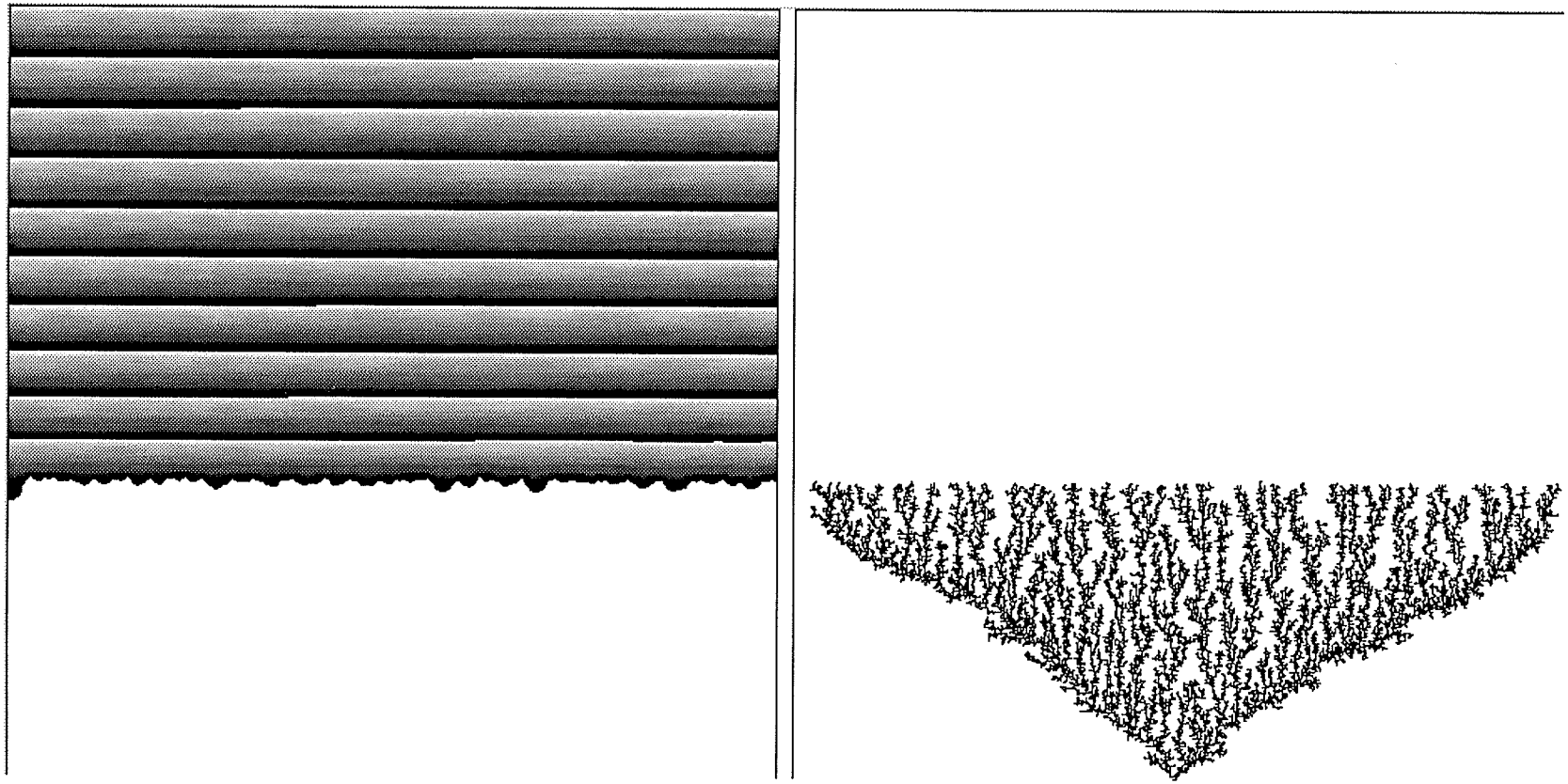
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn9	GS	GS	n/a	Lozi



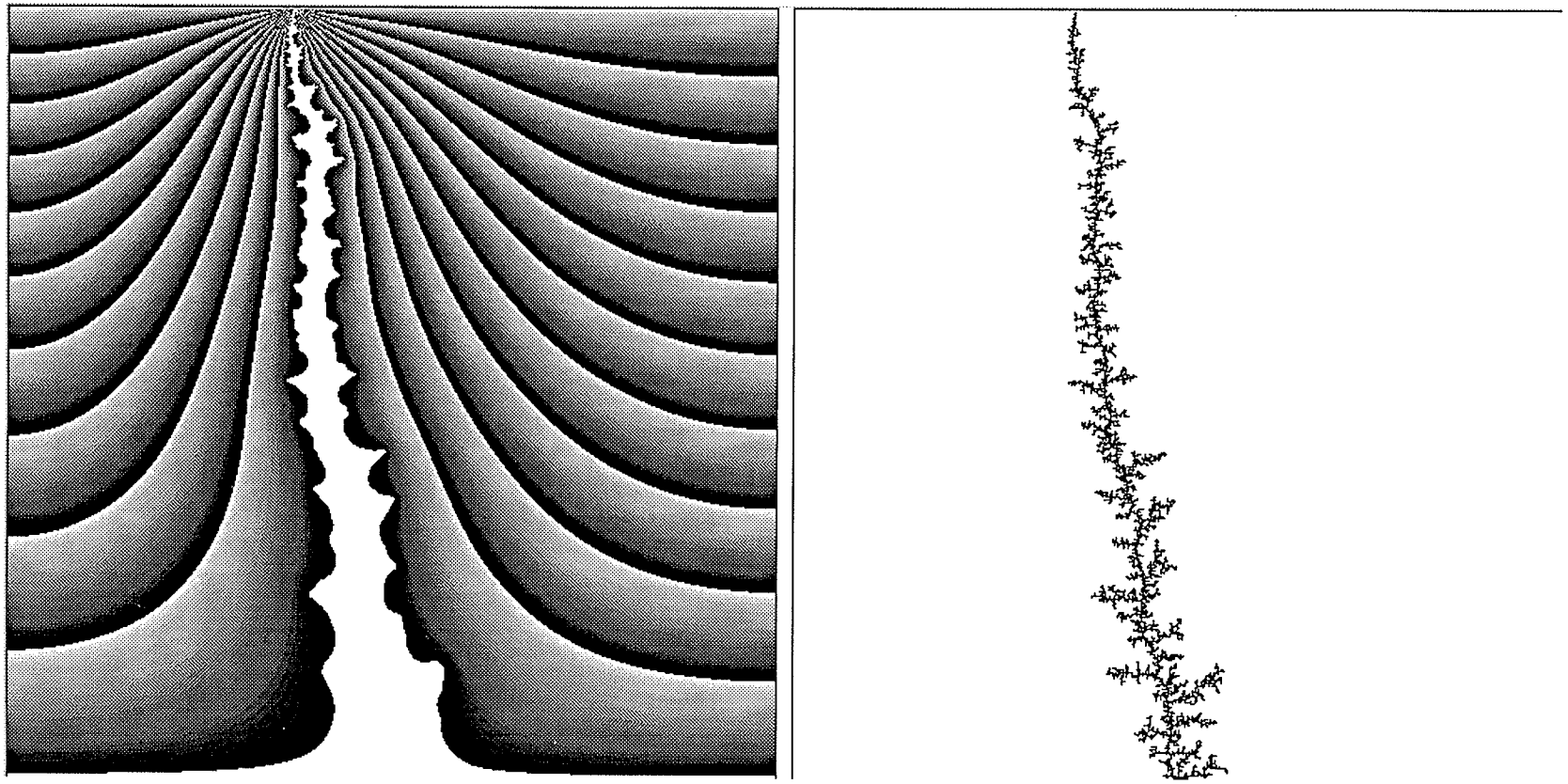
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn10	GS	GS	n/a	Gingerbreadman



Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn11	GS	GS	n/a	Ikeda

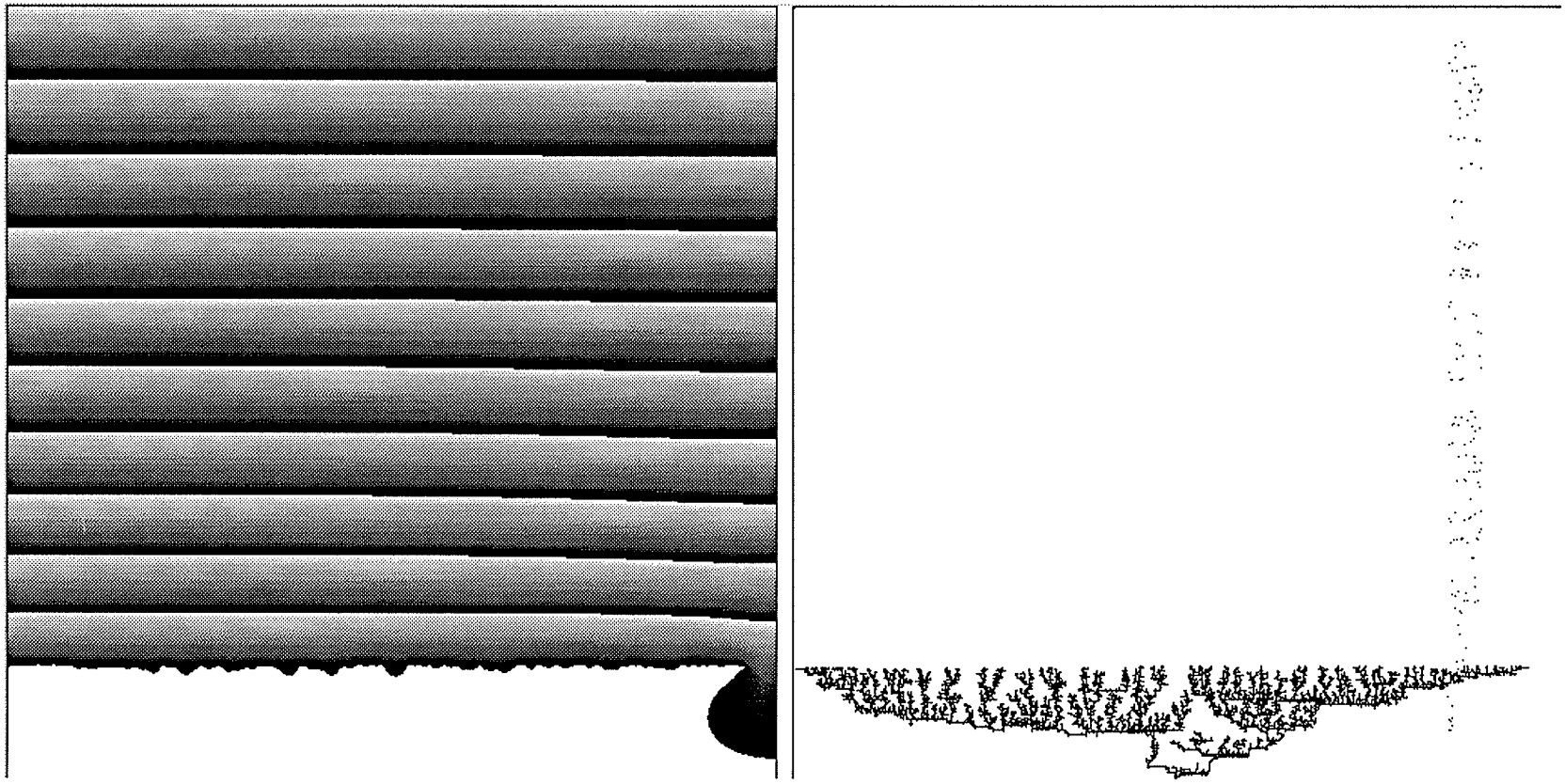


Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn12	GS	GS	n/a	Julia



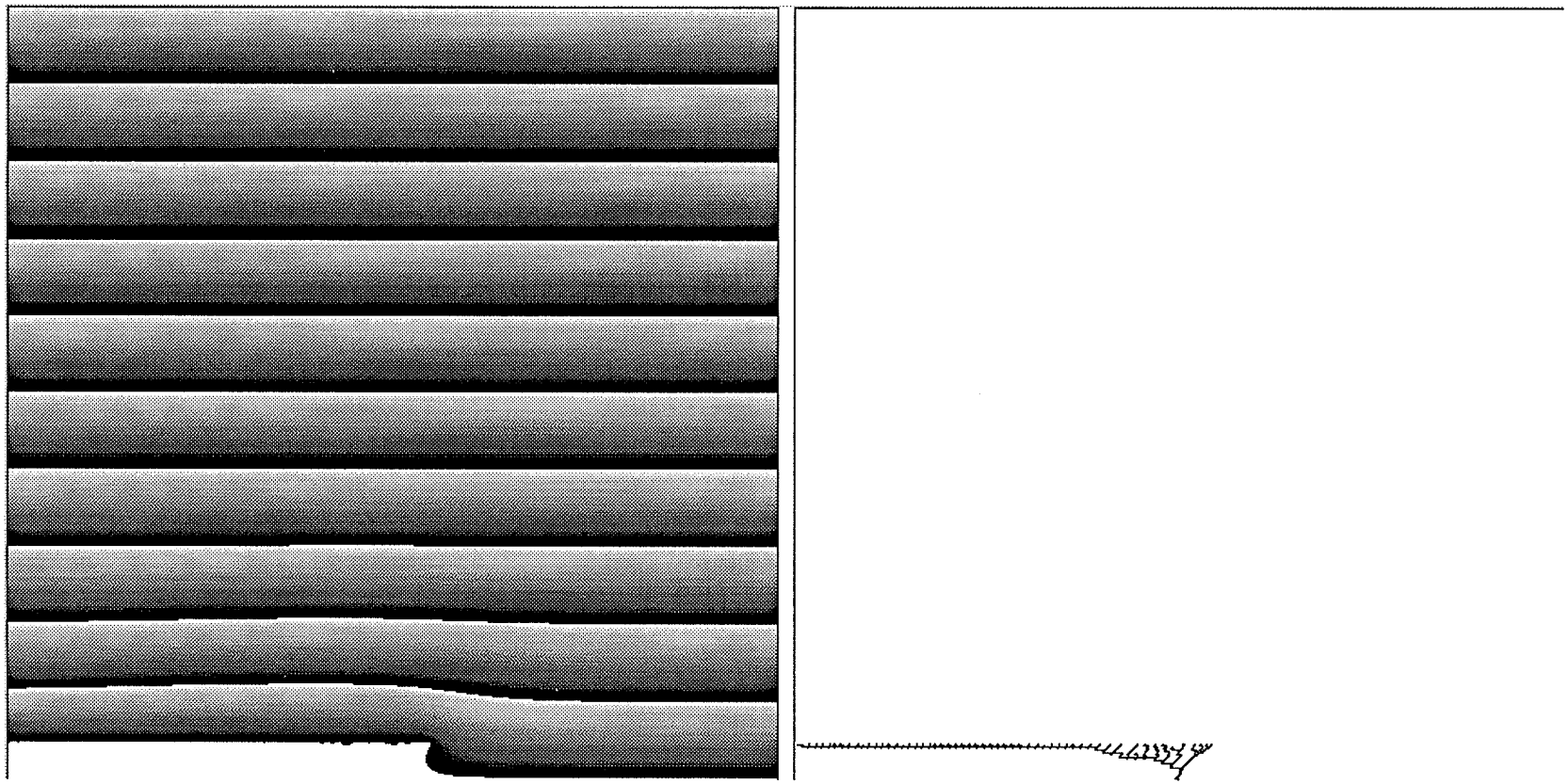
Note: Discharge grew vertically so rapidly, the computed field contours lag far behind.

Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn13	GS	GS	n/a	Lorenz



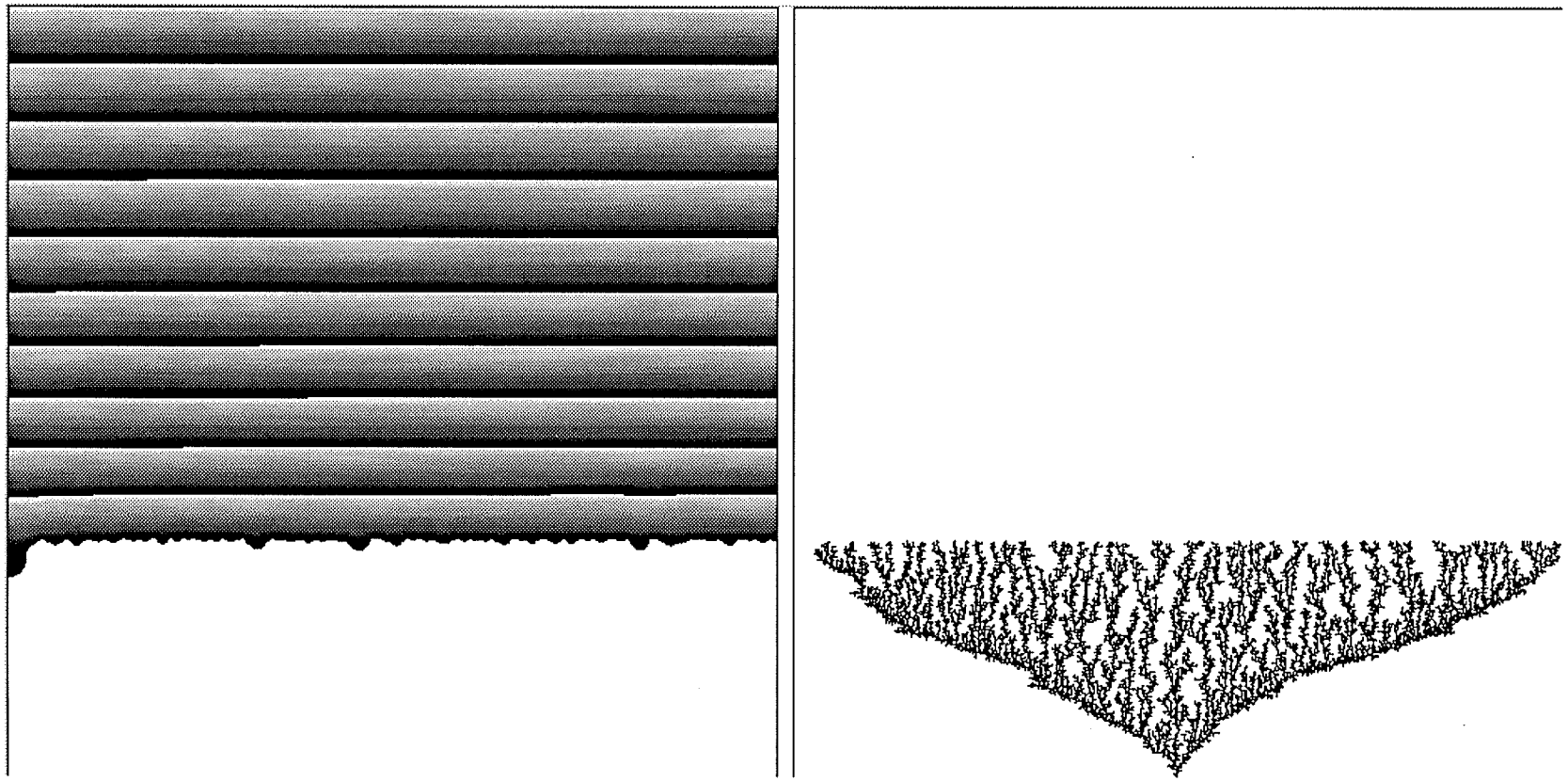
Note: some vertical noise in fractal (right side window). Most likely to a faulty screen capture or accidental over-writing.

Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn14	GS	GS	n/a	Rössler

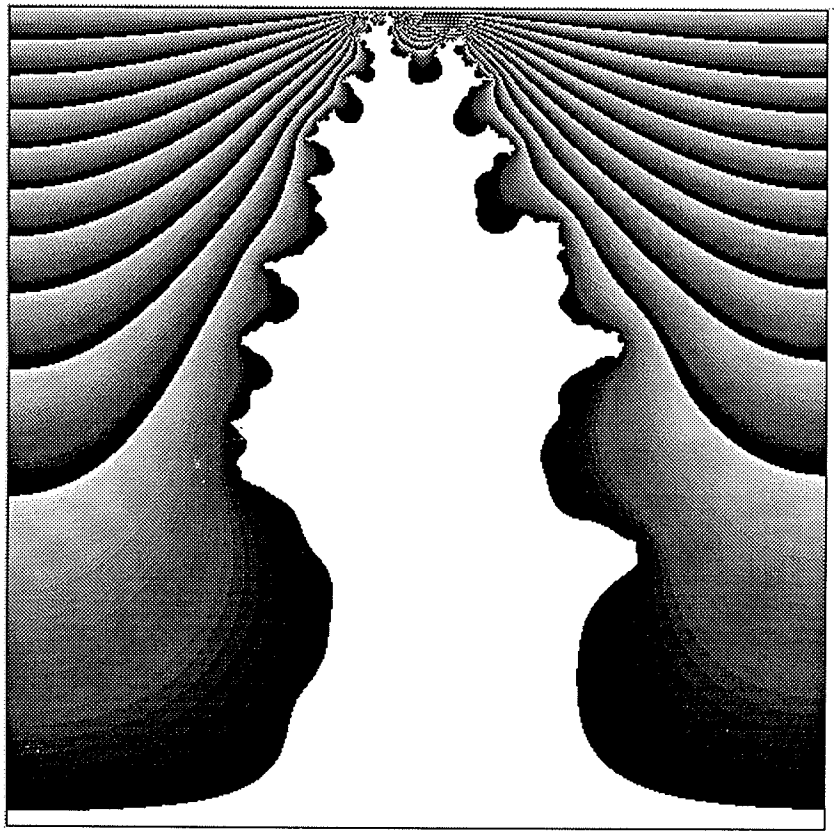


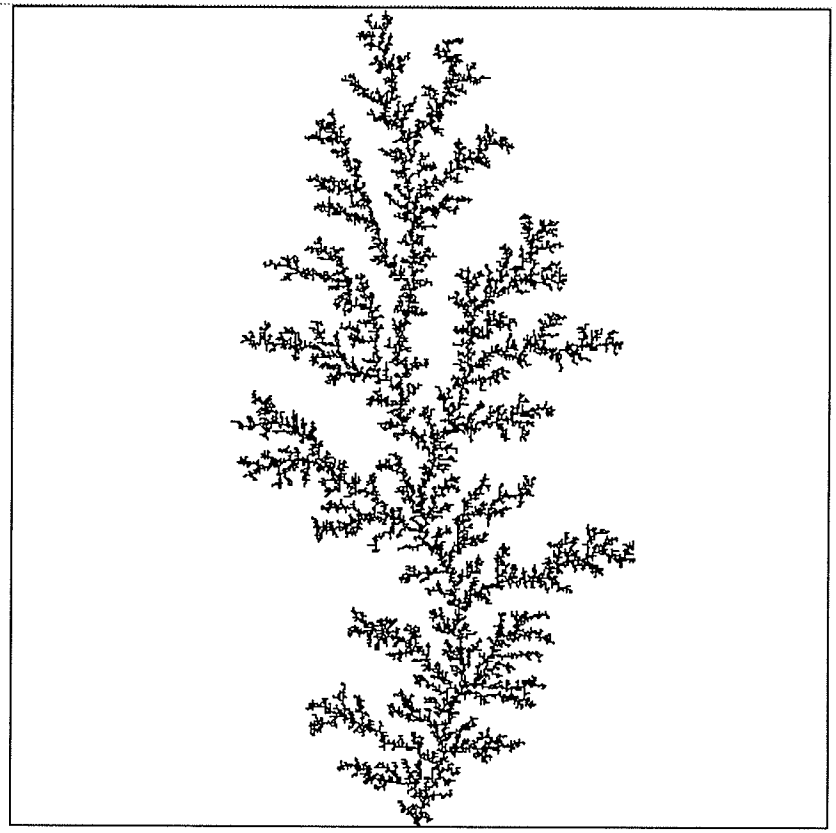
Note: black noise generator. Discharges heads into oncoming left-to-right tag scan.

Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
trfnn15	GS	GS	n/a	Gaussian



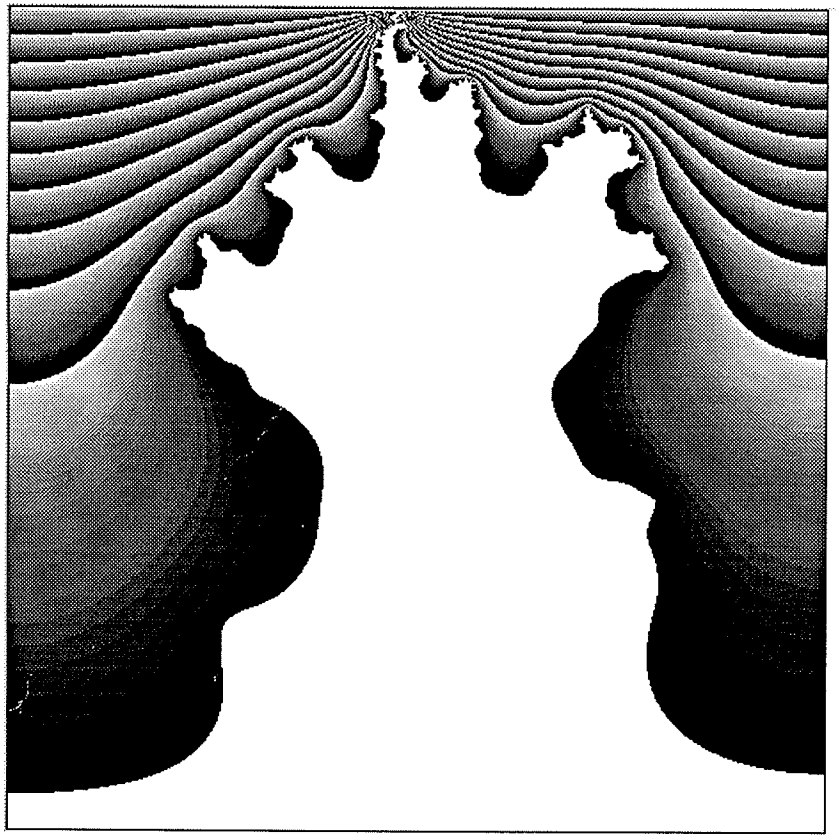
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
app220	GS	GS	n/a	rand2 (81)

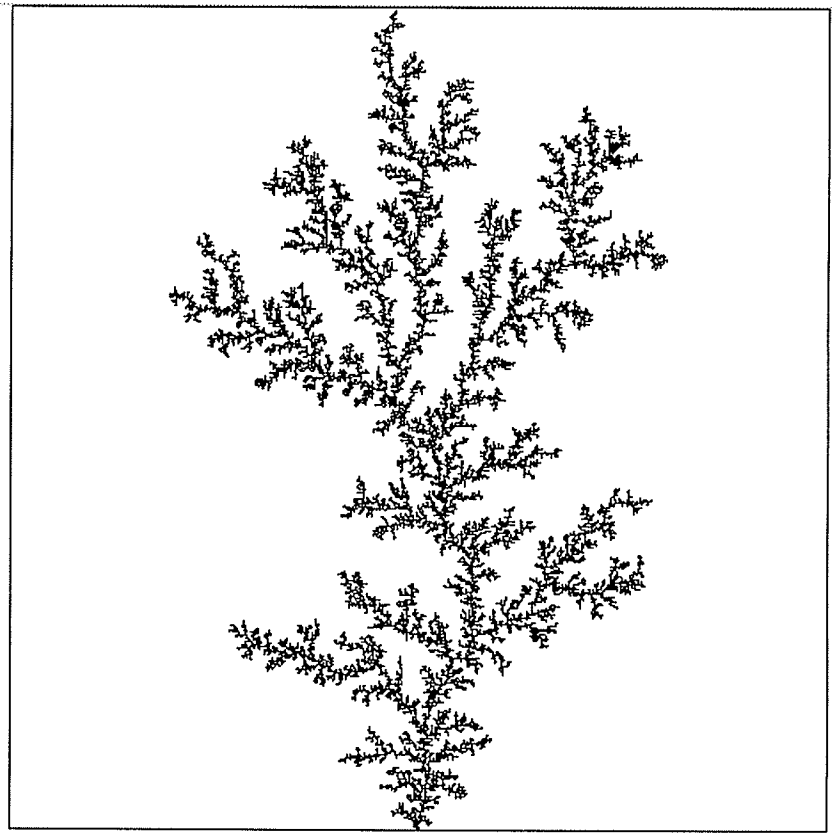




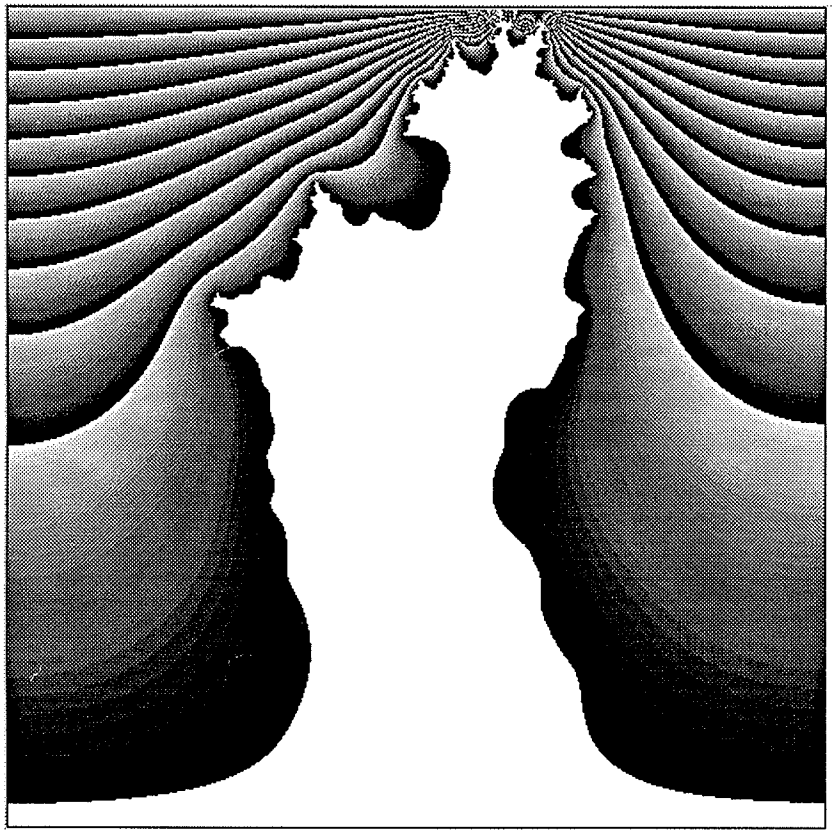
Note: used a 51×51 local update region. Compare with **trfnn4**.

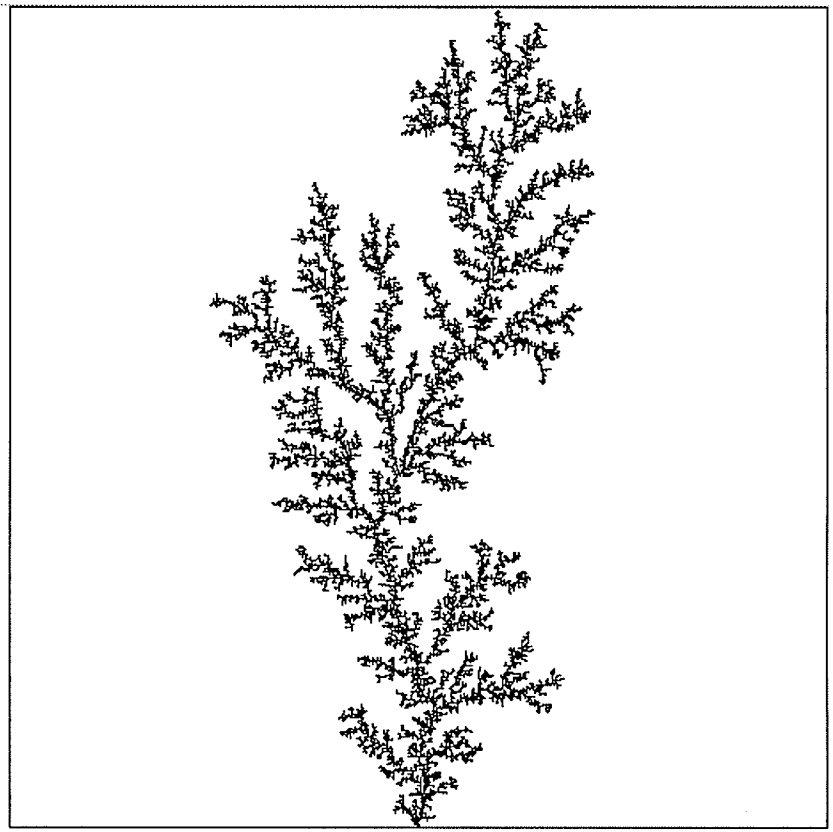
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
app231	GS	SOR	classical	rand2 (81)



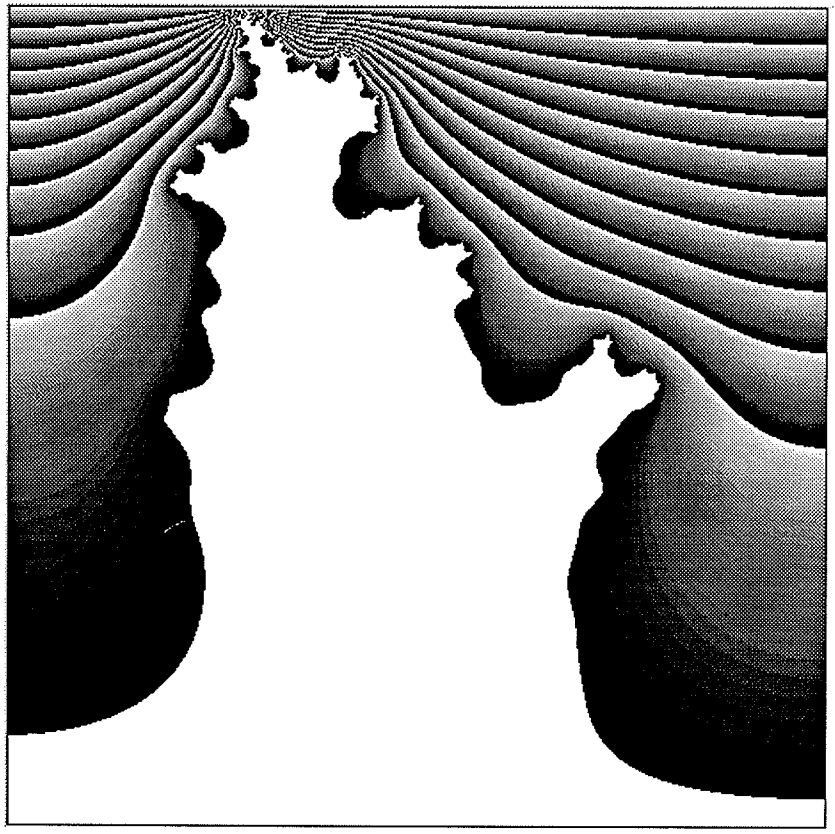


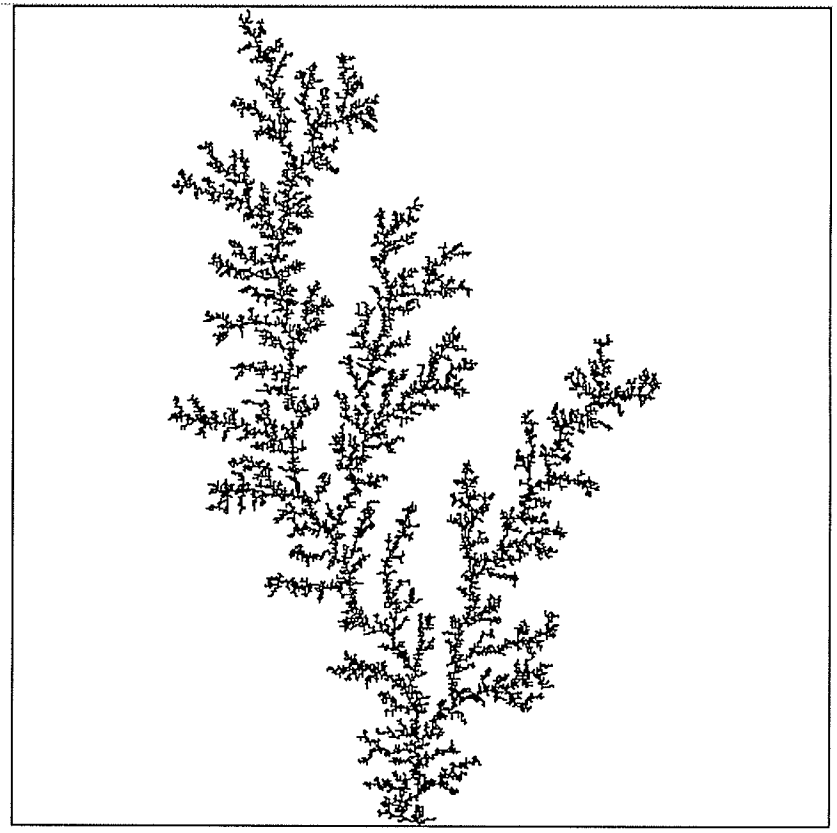
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
ap230	GS	SOR	improved	rand2 (81)



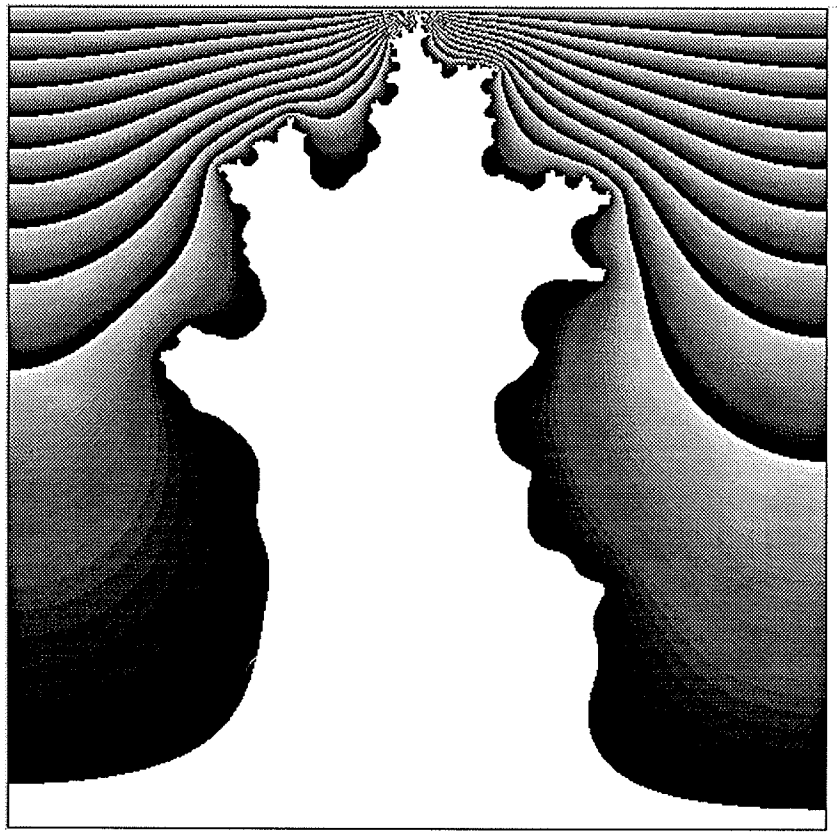


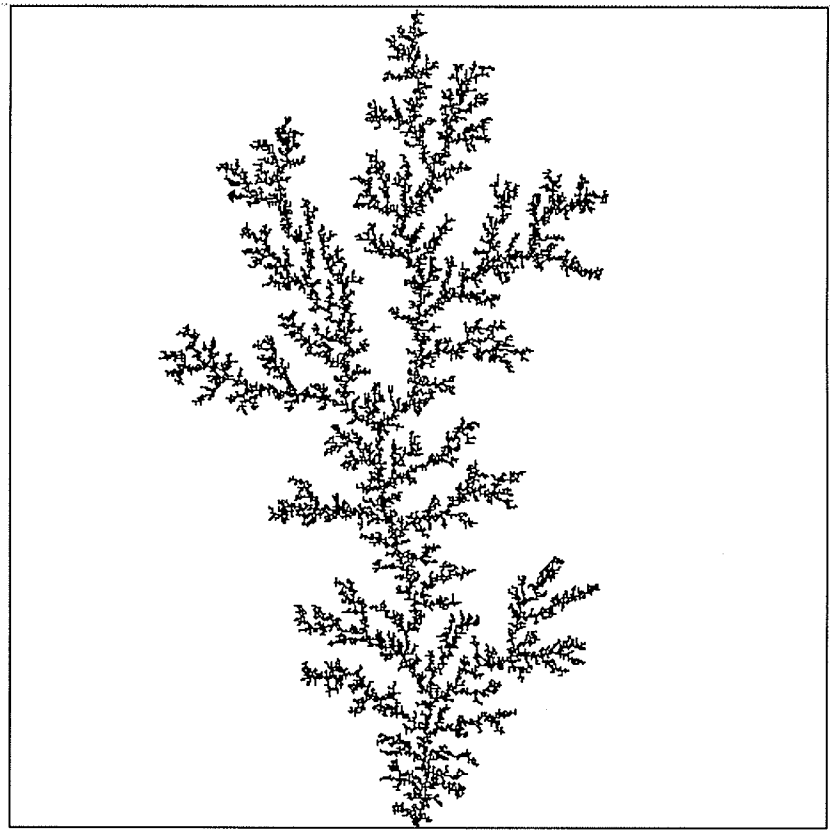
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
app330	SOR	SOR	improved/improved	rand2 (81)



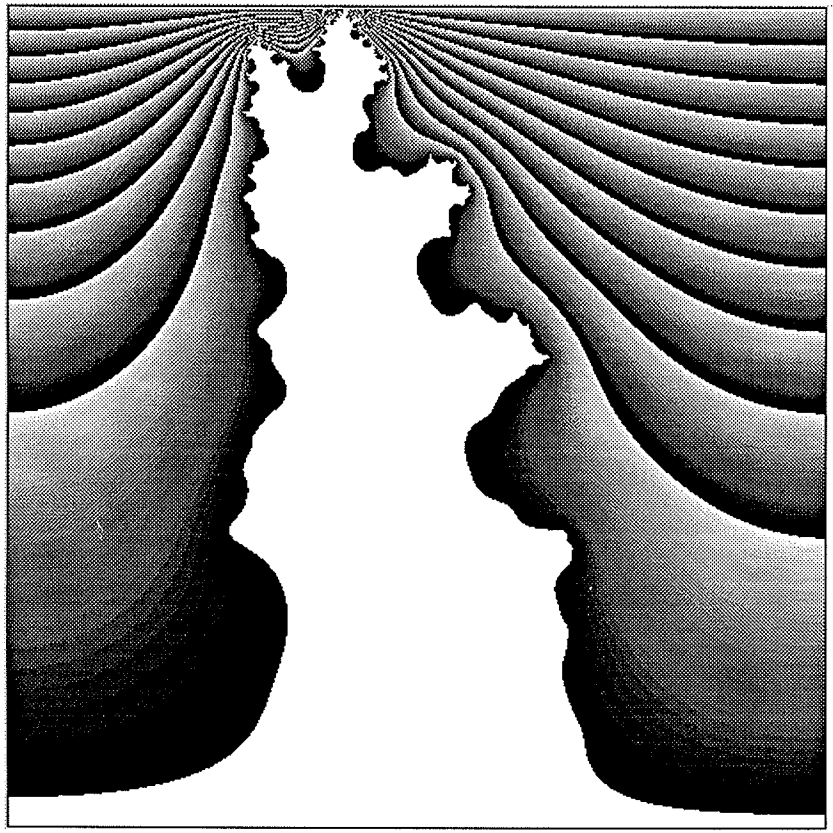


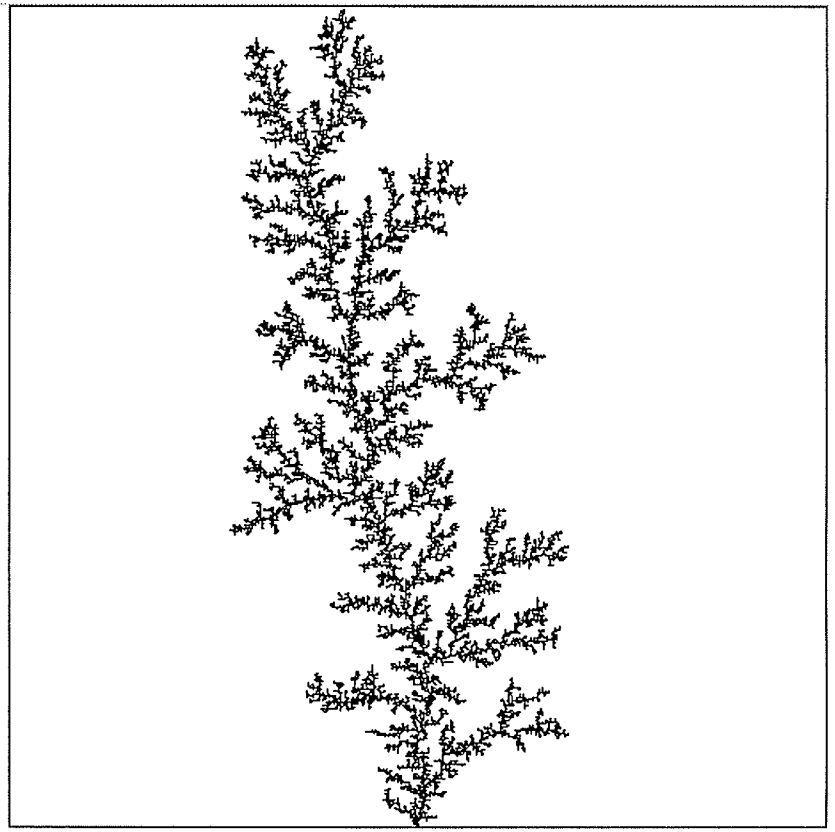
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
treem06	GS	GLOR	n/a	rand2 (81)



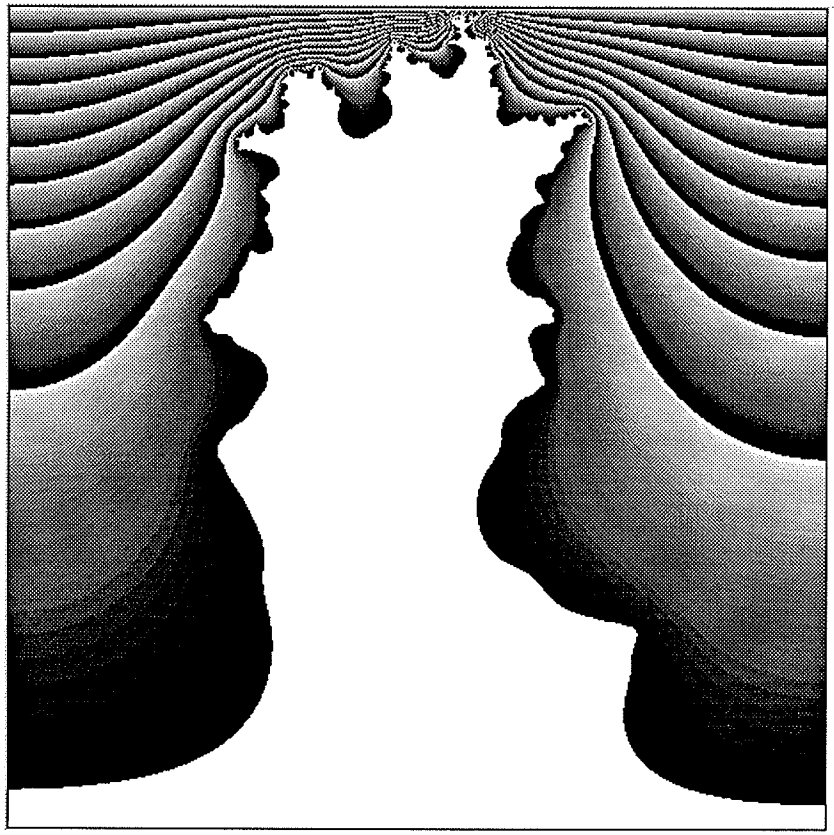


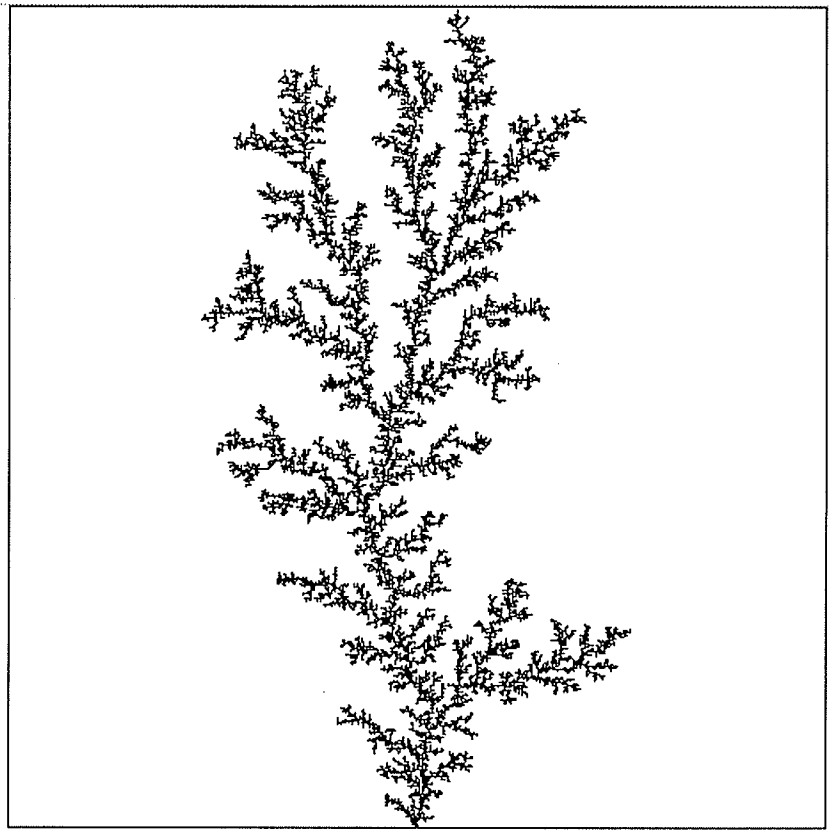
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
treem07	GS	ADGLOR	n/a	rand2 (81)



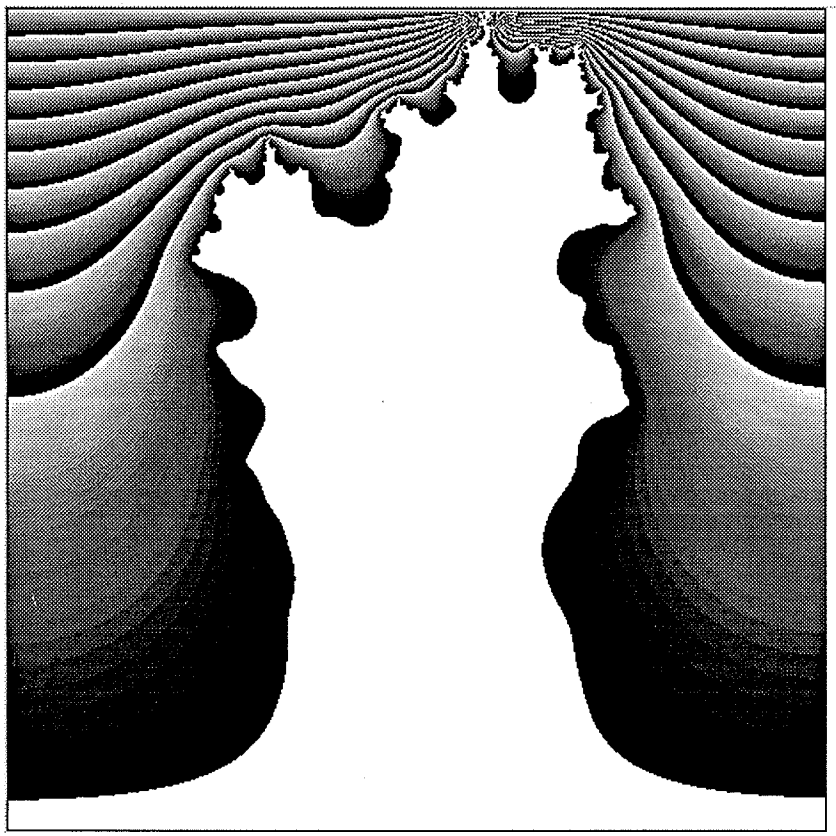


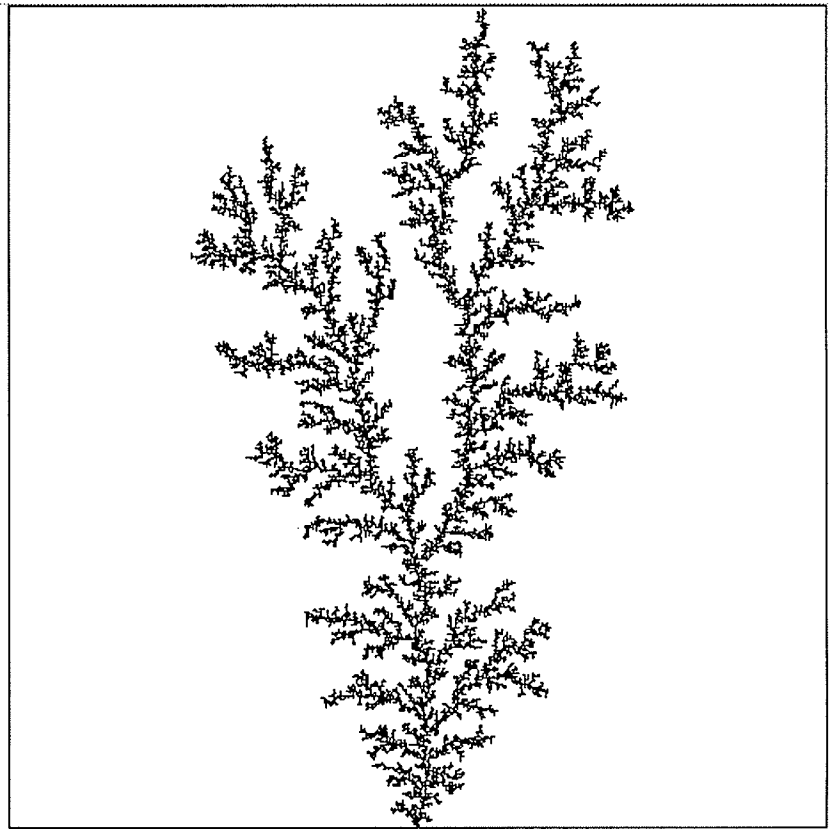
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
app280	GS	SLOR	improved	rand2 (81)



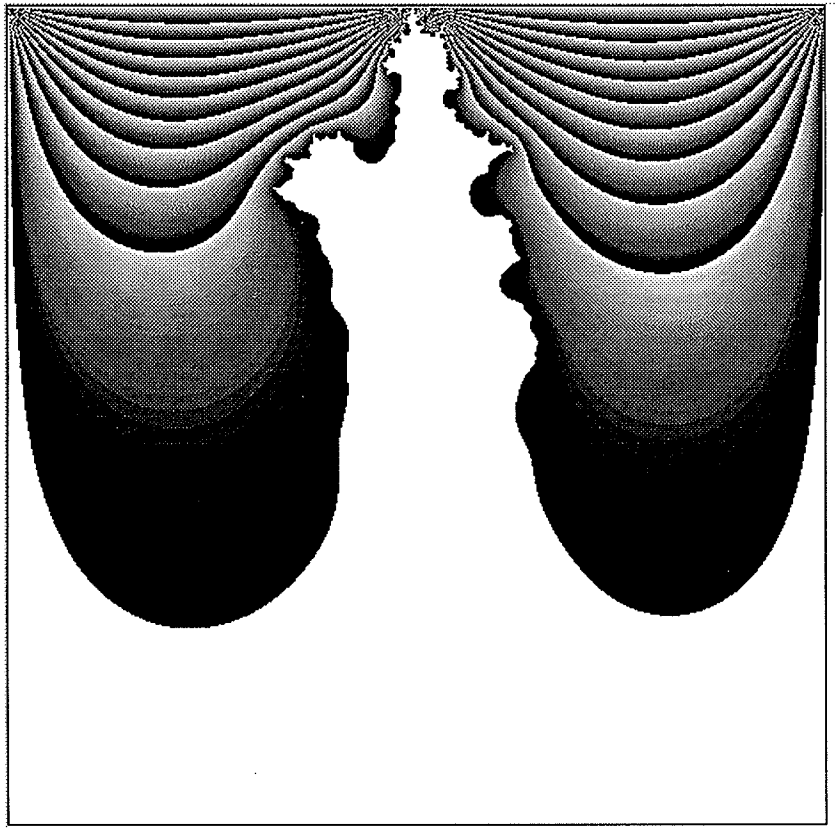


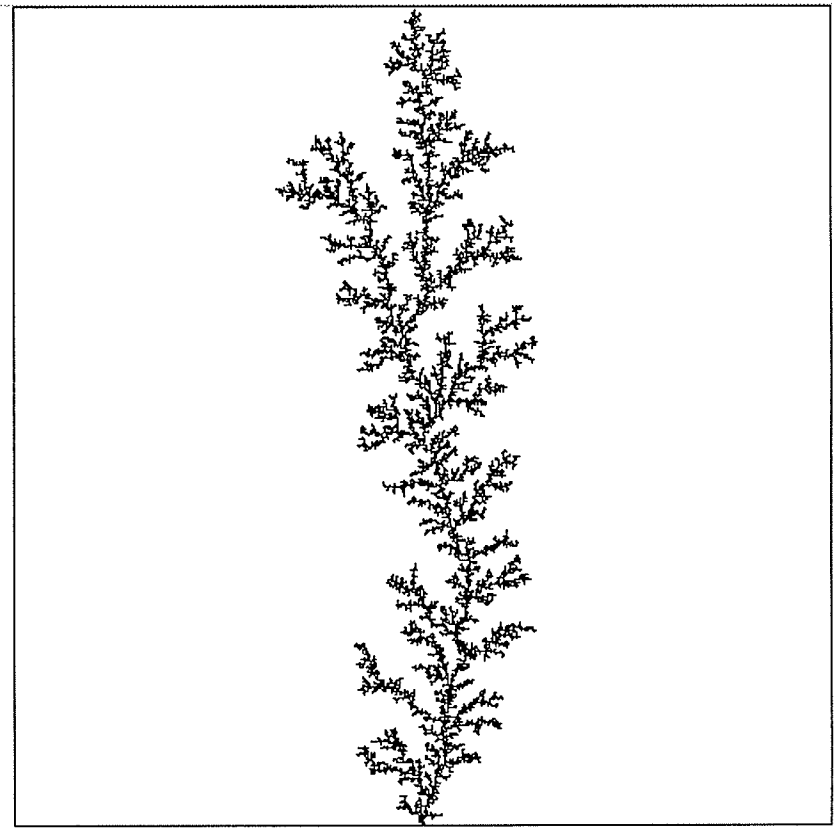
Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
treem12	GS	GBOR	n/a	rand2 (81)





Filename	local iterations	global iterations	λ_{max}, ω_{opt} estimation	random generator
app2d0	GS	SBOR	improved	rand2 (81)





Comments: note the numerical problem in the field alongside the left and right Neumann boundaries due to coding error.