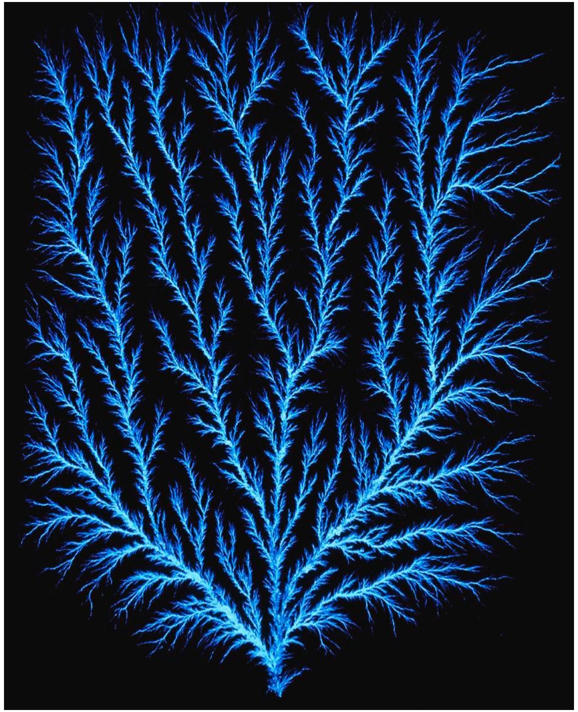
Stochastic electrical tree modeling through the dielectric breakdown model (DBM)

By Justin Burzachiello

Roadmap

1. Electrical trees in nature
2. What is dielectric breakdown?
3. What is the dielectric breakdown model (DBM)?
4. Overview of my DBM simulation of electrical treeing
5. 4 different visualizations of electrical tree evolution:
    a. Tree structure
    b. Probability distribution
    c. Electrostatic scalar potential
    d. Electric field
6. Basic gyration properties of the structure.
7. How other models stem from DBM
    a. Diffusion limited aggregation model, Eden model, …

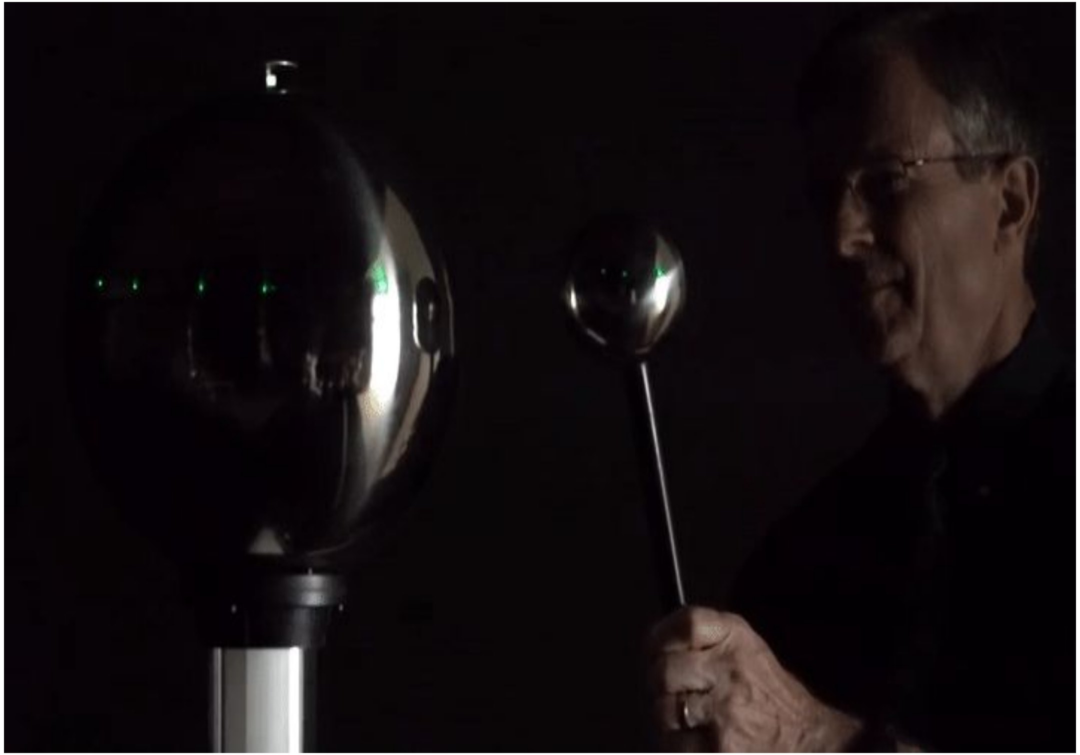
Electrical Trees



Dielectric breakdown

The sudden failure of an insulating material, typically due to the exceeding of its electric field strength, leading to the rapid increase in electrical conductivity.

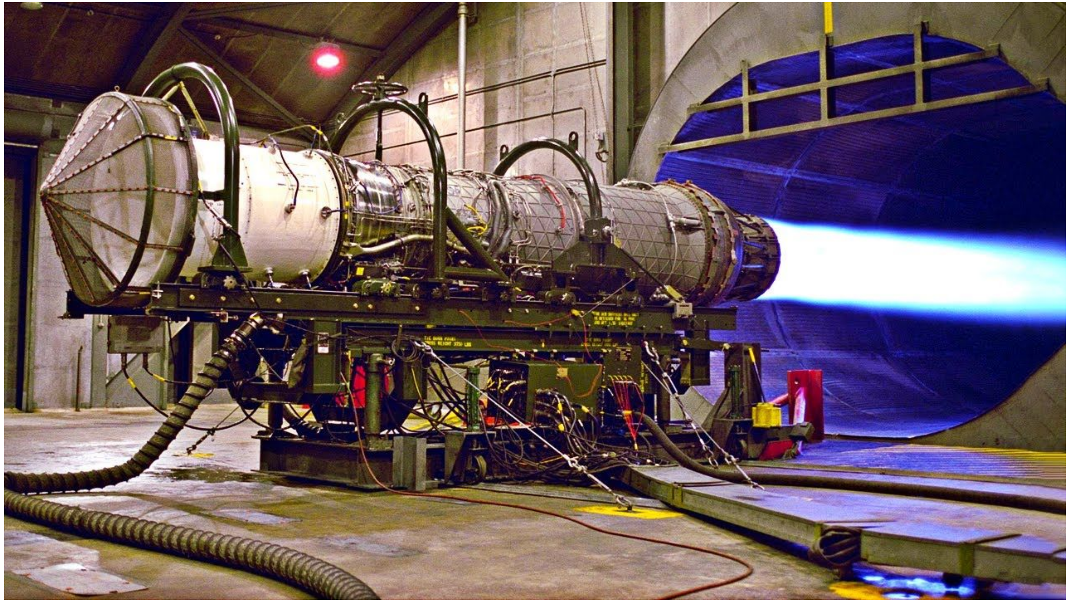
> Essentially, a large enough voltage across an insulator can cause it to “break down”, enabling current to flow through it.
>
> Example: Cloud-to-ground lightning occurs when the voltage across the insulative air between the bottom of the cloud and the ground below it is high enough for some of the air to conduct electrical current.



Dielectric breakdown

The sudden failure of an insulating material, typically due to the exceeding of its electric field strength, leading to the rapid increase in electrical conductivity.

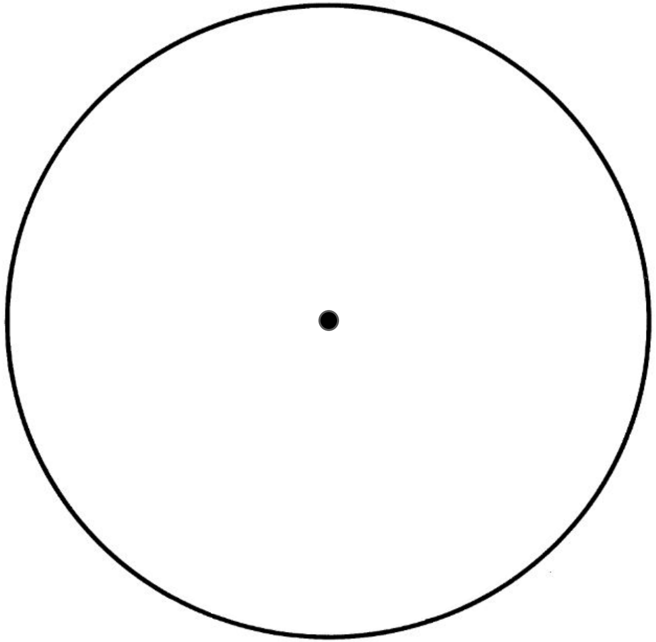
> Essentially, a large enough voltage across an insulator can cause it to “break down”, enabling current to flow through it.
>
> Example: Cloud-to-ground lightning occurs when the voltage across the insulative air between the bottom of the cloud and the ground below it is high enough for some of the air to conduct electrical current.



Dielectric breakdown model (DBM)

A computational algorithm used to stochastically generate electrical trees through the following 7 steps:

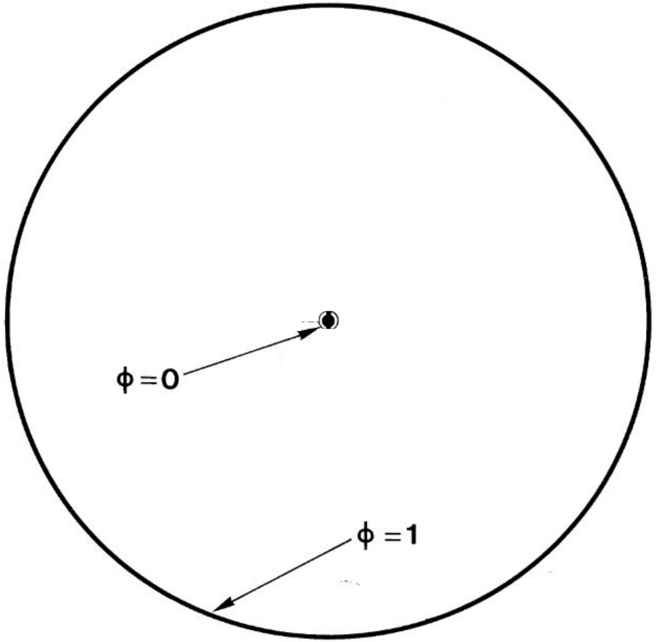
1. Construct a finite grid with a single cell representing the seed of the tree



Dielectric breakdown model (DBM)

A computational algorithm used to stochastically generate electrical trees through the following 7 steps:

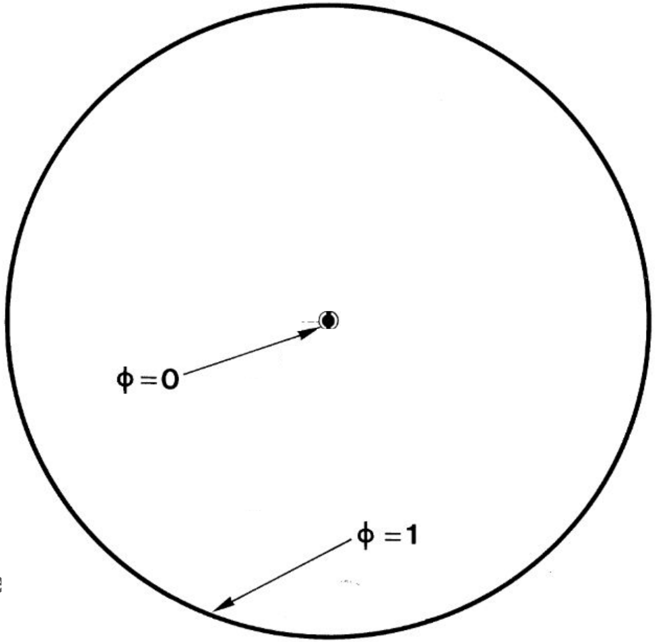
1. Construct a finite grid with a single cell representing the seed of the tree
2. Initialize the potential field's boundaries across the domain of the grid:
    - the electrical tree is one boundary; the source of high potential is another



Dielectric breakdown model (DBM)

A computational algorithm used to stochastically generate electrical trees through the following 7 steps:

1. Construct a finite grid with a single cell representing the seed of the tree
2. Initialize the potential field's boundaries across the domain of the grid:
    - the electrical tree is one boundary; the source of high potential is another
3. Solve the electrostatic Laplace PDE numerically on this grid to compute the potential at each coordinate



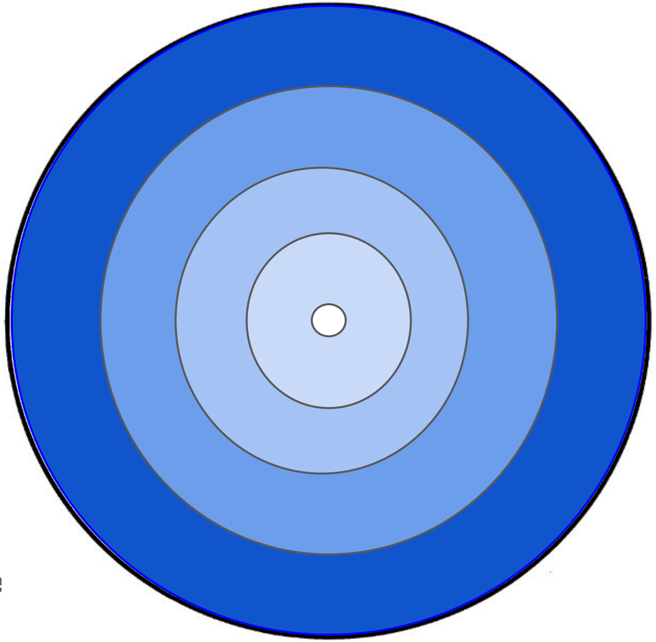
$$\nabla^2 \phi = 0$$

$$\phi_{i,k} = \frac{1}{4}\left(\phi_{i+1,k} + \phi_{i-1,k} + \phi_{i,k+1} + \phi_{i,k-1}\right)$$

Dielectric breakdown model (DBM)

A computational algorithm used to stochastically generate electrical trees through the following 7 steps:

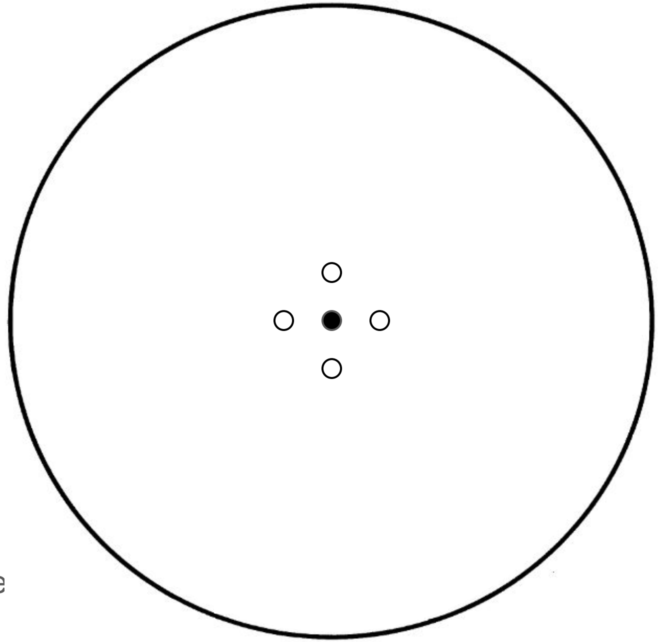
1. Construct a finite grid with a single cell representing the seed of the tree
2. Initialize the potential field's boundaries across the domain of the grid:
   - the electrical tree is one boundary; the source of high potential is another
3. Solve the electrostatic Laplace PDE numerically on this grid to compute the potential at each coordinate



Dielectric breakdown model (DBM)

A computational algorithm used to stochastically generate electrical trees through the following 7 steps:

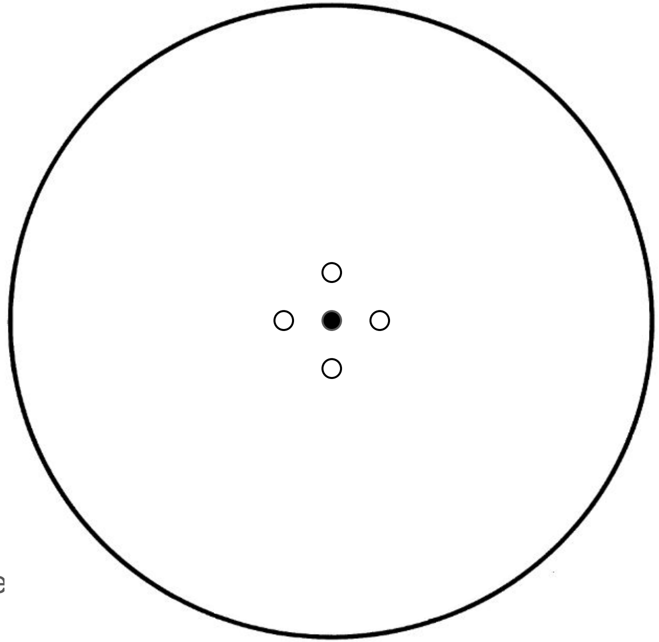
1. Construct a finite grid with a single cell representing the seed of the tree
2. Initialize the potential field’s boundaries across the domain of the grid:
    - the electrical tree is one boundary; the source of high potential is another
3. Solve the electrostatic Laplace PDE numerically on this grid to compute the potential at each coordinate
4. Obtain the coordinates of each cell adjacent to the tree.



Dielectric breakdown model (DBM)

A computational algorithm used to stochastically generate electrical trees through the following 7 steps:

1. Construct a finite grid with a single cell representing the seed of the tree
2. Initialize the potential field's boundaries across the domain of the grid:
    - the electrical tree is one boundary; the source of high potential is another
3. Solve the electrostatic Laplace PDE numerically on this grid to compute the potential at each coordinate
4. Obtain the coordinates of each cell adjacent to the tree.
5. Assign probabilities to each of the cells from (4)

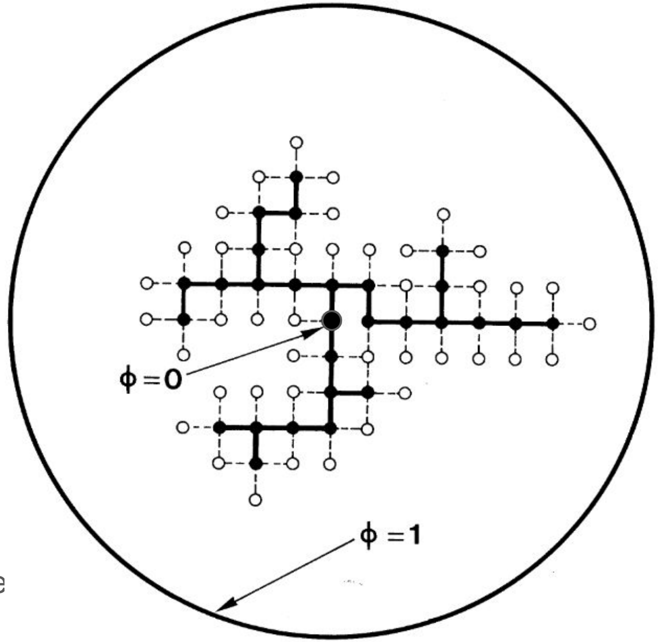


$$p(i,k \rightarrow i',k') = \frac{(\phi_{i',k'})^{\eta}}{\sum(\phi_{i',k'})^{\eta}}$$

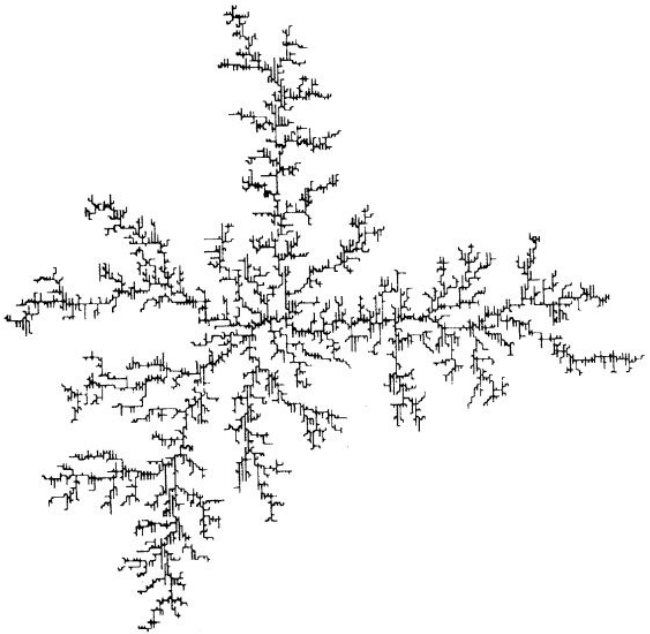
Dielectric breakdown model (DBM)

A computational algorithm used to stochastically generate electrical trees through the following 7 steps:

1. Construct a finite grid with a single cell representing the seed of the tree
2. Initialize the potential field's boundaries across the domain of the grid:
   - the electrical tree is one boundary; the source of high potential is another
3. Solve the electrostatic Laplace PDE numerically on this grid to compute the potential at each coordinate
4. Obtain the coordinates of each cell adjacent to the tree.
5. Assign probabilities to each of the cells from (4)
6. Add a cell to the tree by choosing one according to the probability distribution created in (5)
7. Repeat steps (3) - (6) for as many cells as desired



Dielectric breakdown model (DBM)



A computational algorithm used to stochastically generate electrical trees through the following 7 steps:

1. Construct a finite grid with a single cell representing the seed of the tree
2. Initialize the potential field's boundaries across the domain of the grid:
   - the electrical tree is one boundary; the source of high potential is another
3. Solve the electrostatic Laplace PDE numerically on this grid to compute the potential at each coordinate
4. Obtain the coordinates of each cell adjacent to the tree.
5. Assign probabilities to each of the cells from (4)
6. Add a cell to the tree by choosing one according to the probability distribution created in (5)
7. Repeat steps (3) - (6) for as many cells as desired

How I implemented the DBM

I used Python to implement the DBM. Doing so involved 5 steps:

1. Numerically solving the Laplace equation on a 2-dimensional grid by convolution.
    a. Convergence was obtained once the mean of the elementwise difference in potential values between the ith iterate of the potential field and the (i-1)th became less than some threshold (usually 1e-6)

```python
def initialize_grid(N, boundary_val):
    #create NxN array of empty tuples
    grid = np.empty((N, N), dtype=np.dtype([('potential', float), ('is_tree', bool)]))

    #initialize values of grid
    # grid['potential'] = create_radial_array(N, 0, boundary_val)
    grid['potential'] = 0.0
    grid['is_tree'] = False

    #create seed of the electrical tree in the center of the grid
    grid[N//2, N//2] = (0.0, True)

    #set boundary conditions
    grid[0, :]['potential']      = boundary_val  # Top boundary
    grid[-1, :]['potential']     = boundary_val  # Bottom boundary
    grid[:, 0]['potential']      = boundary_val  # Left boundary
    grid[:, -1]['potential']     = boundary_val  # Right boundary

    # plt.imshow(grid['potential'])
    # plt.show()
```

```python
def convolve_2d(grid):
    convolved_grid = np.copy(grid)
    height, width = grid.shape

    for r in range(1, height-1):
        for c in range(1, width-1):
            if not grid[r, c]['is_tree']:
                #implicitly uses Laplacian kernel: ([[0, 1/4, 0],[1/4, 0, 1/4],[0, 1/4, 0]])
                adjacent_potentials = np.array((
                    grid[r+1,c]['potential'],
                    grid[r-1,c]['potential'],
                    grid[r,c+1]['potential'],
                    grid[r,c-1]['potential']
                ))
                convolved_grid[r, c]['potential'] = np.mean(adjacent_potentials)

    return convolved_grid
```

```python
def solve_laplace(grid, threshold):
    error = 1e6
    while error > threshold:
        convolved_grid = convolve_2d(grid)
        error = np.mean(np.square((grid['potential'] - convolved_grid['potential'])))
        grid = convolved_grid

    return grid
```

How I implemented the DBM

I used Python to implement the DBM. Doing so involved 5 steps:

1. Numerically solving the Laplace equation on a 2-dimensional grid by convolution.
    a. Convergence was obtained once the mean of the elementwise difference in potential values between the ith iterate of the potential field and the (i-1)th became less than some threshold (usually 1e-6)
2. Computing the probability distribution of possible new cell sites based on the power-law equation governing DBM.

```python
def grow_tree(grid, eta, frames_prob, i):
    rows, cols = grid.shape
    probs = np.zeros((rows, cols))
    new_grid = grid.copy()

    possible_growth_coords = set()
    for r in range(1, rows-1):
        for c in range(1, cols-1):
            if grid[r, c]['is_tree']:
                for adj_cell in ((r+1, c), (r-1, c), (r, c+1), (r, c-1)):
                    if not grid[adj_cell]['is_tree']:
                        possible_growth_coords.add(adj_cell)

    potentials_at_possibles = np.array([grid[coord]['potential']**eta for coord in possible_growth_coords])
    total_potential_at_possibles = np.sum(potentials_at_possibles)

    for possible_coord in possible_growth_coords:
        probs[possible_coord] = (grid[possible_coord]['potential']**eta) / total_potential_at_possibles
    frames_prob[i,:,:] =        np.copy(probs)

    coord_of_tree_growth = choose_coordinate(probs)
    new_grid[coord_of_tree_growth] = (0.0, True)#, False)

    return new_grid
```

How I implemented the DBM

I used Python to implement the DBM. Doing so involved 5 steps:

1. Numerically solving the Laplace equation on a 2-dimensional grid by convolution.
   a. Convergence was obtained once the mean of the elementwise difference in potential values between the ith iterate of the potential field and the (i-1)th became less than some threshold (usually 1e-6)
2. Computing the probability distribution of possible new cell sites based on the power-law equation governing DBM.
3. Visualizing the evolution of the tree, the probability distribution, the potential field, and the contour plot of the potential field (the electric field).

How I implemented the DBM

I used Python to implement the DBM. Doing so involved 5 steps:

1. Numerically solving the Laplace equation on a 2-dimensional grid by convolution.
   a. Convergence was obtained once the mean of the elementwise difference in potential values between the ith iterate of the potential field and the (i-1)th became less than some threshold (usually 1e-6)
2. Computing the probability distribution of possible new cell sites based on the power-law equation governing DBM.
3. Visualizing the evolution of the tree, the probability distribution, the potential field, and the contour plot of the potential field (the electric field).
4. Comparing trees generated for different values of eta:

```python
def run_dbm_(num_tree_cells, N, threshold, boundary_val, eta):
    grid = initialize_grid(N, boundary_val)
    convolved_grid = solve_laplace(grid, threshold)

    radii_of_gyration = []
    new_grid = None
    for i in range(1, num_tree_cells):
        new_grid = grow_tree(convolved_grid, eta, frames_prob, i)
        convolved_grid = solve_laplace(new_grid, threshold)

        rg = get_rg(new_grid['is_tree'])
        gt = get_gyration_tensor(new_grid['is_tree'])
        moments = get_principle_moments(gt)
        num_cells = np.sum(new_grid['is_tree'])
        radius_of_gyration = get_radius_of_gyration(moments, num_cells)
        acylindricity = get_acylindricity(moments)
        radii_of_gyration.append(rg)

    return convolved_grid['is_tree']
```

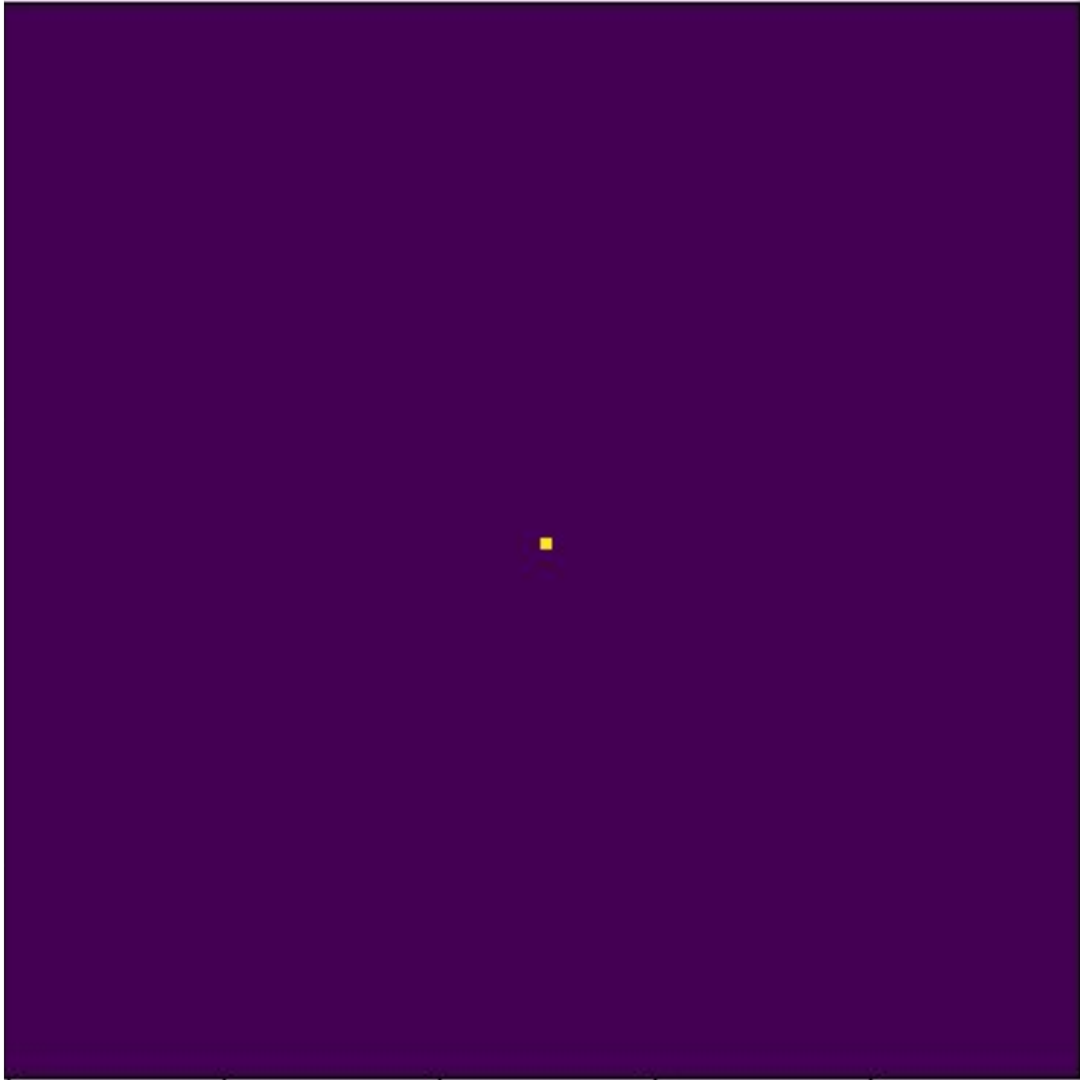
```python
num_tree_cells = 40
N = 100
threshold = 1e-6
boundary_val = 1

for eta in (0, 0.5, 1, 2):
    is_tree = run_dbm(num_tree_cells, N, threshold, boundary_val, eta)
```

Evolution of tree structure

Fractal - A mathematical structure / set which exhibits scale invariance (i.e. the subsets are similar to the whole – self-similarity)

This structure exhibits **self-similarity** since the branches have branches which have branches and so on

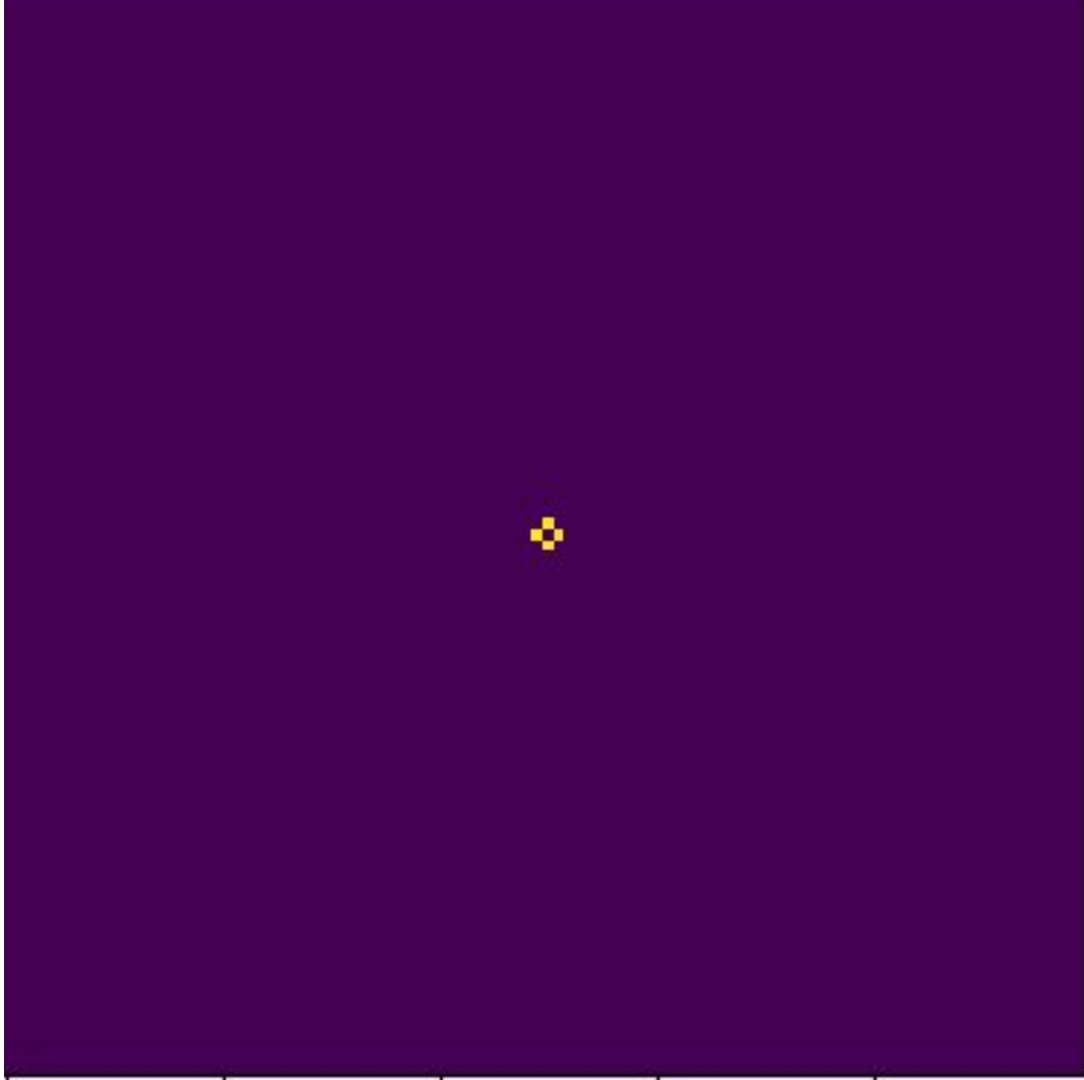
The electrical tree is thus a fractal. It’s fractal / Hausdorff dimension is around 1.72.



Evolution of probability distribution

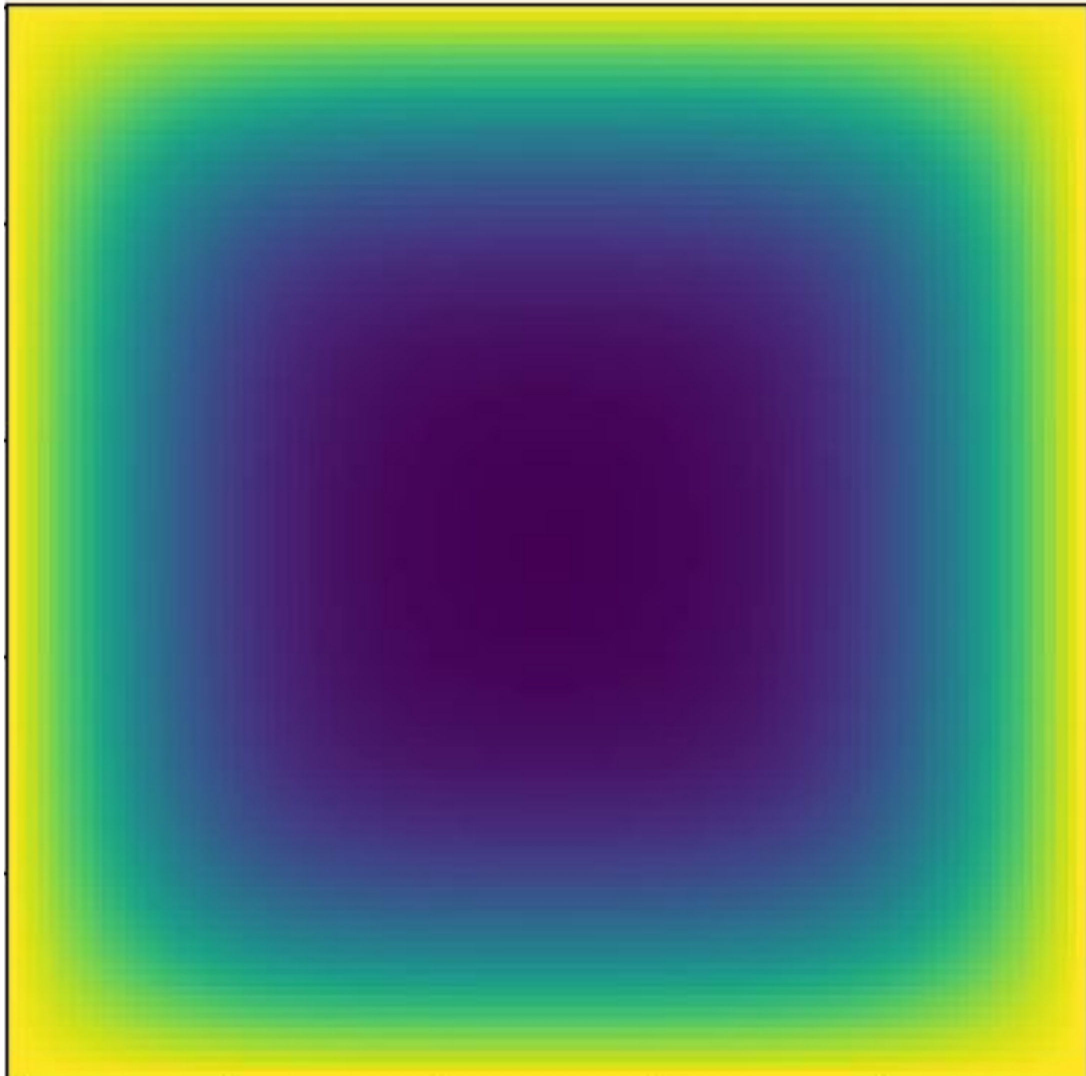
Faraday cage - An enclosure made of conducting material used to block electromagnetic influence

The conducting regions which broke down due to dielectric breakdown create a Faraday cage which blocks electromagnetic fields from entering (prohibiting additional breakdown from within the structure)



Evolution of electrostatic scalar potential

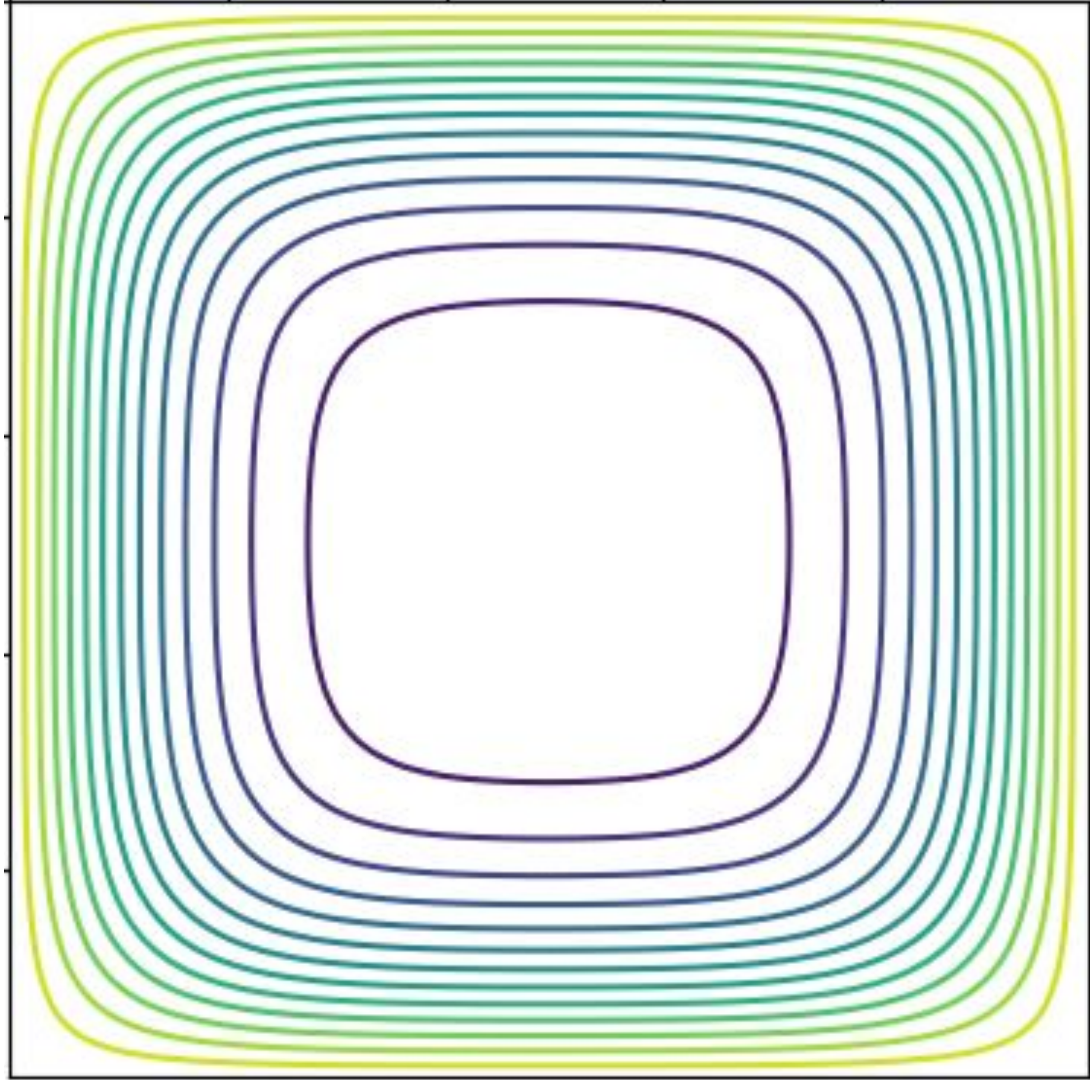
The structure of the electrical tree at one instance of time alters the electrostatic environment, thus changing the solution to the Laplace equation and the probability field for the next instance of time.



Evolution of electric field

Tip effect - The build-up of charge at sharp points along a conductor, resulting in strong electric field at that point

The tips of the electrical tree act as tips of a conductor, and the contour plots bunching up shows that the electric field is relatively strong at these points.

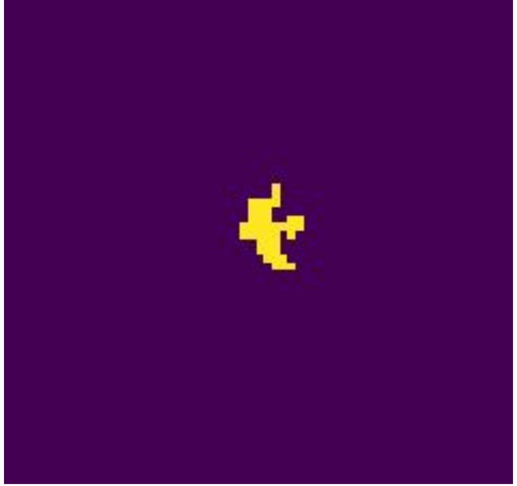


Structure of a tree

Radius of gyration – The RMS distance of all points from center of mass

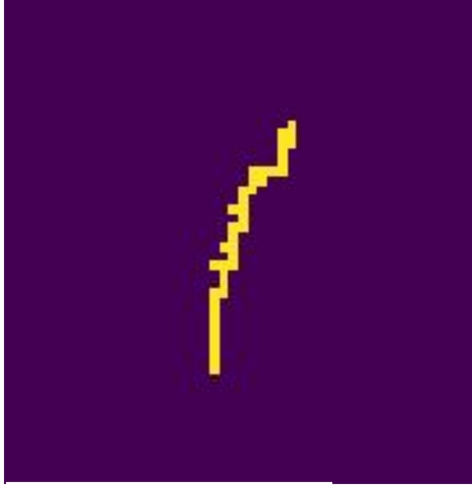
Acylindricity – A measure of how “cylindrical” a structure is

Gyration tensor – A tensor which describes the second moments of position for a collection of particles



radius of gyration: 3.11

acylindricity: 3.57



radius of gyration: 7.81

acylindricity: 59.11

$$S_{ij} = \frac{1}{N}\sum_{k=1}^{N}\left(r_i^{(k)} - r_i^{(CM)}\right)\left(r_j^{(k)} - r_j^{(CM)}\right); \quad S = \begin{pmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{pmatrix}$$

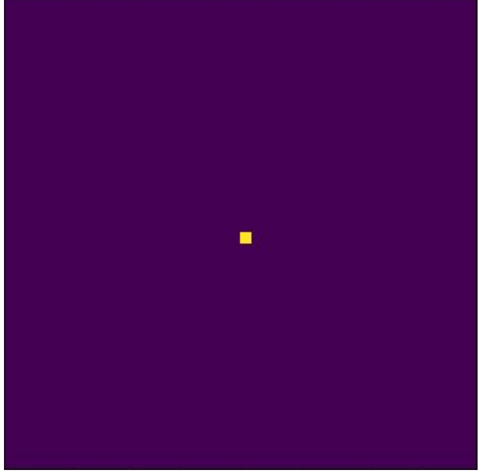
$$\mathrm{c} = L2 - L1$$

$$R_{\mathrm{g}}^2 \stackrel{\mathrm{def}}{=} \frac{1}{N}\sum_{k=1}^{N}\left|\mathbf{r}_k - \mathbf{r}_{\mathrm{mean}}\right|^2$$

How DBM generalizes to other models

eta = 1: Diffusion limited aggregation

Linearly proportional to potential strength



1

0.5

$$p(i,k \rightarrow i',k') = \frac{(\phi_{i',k'})^{\eta}}{\sum(\phi_{i',k'})^{\eta}}$$

0

2

How DBM generalizes to other models

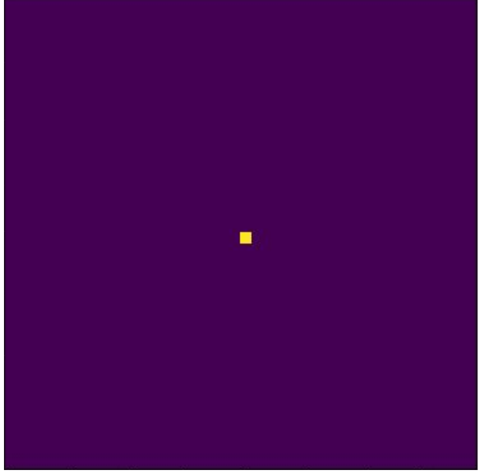
eta = 1: Diffusion limited aggregation

Linearly proportional to potential strength

eta = 0: Eden model of cancer growth

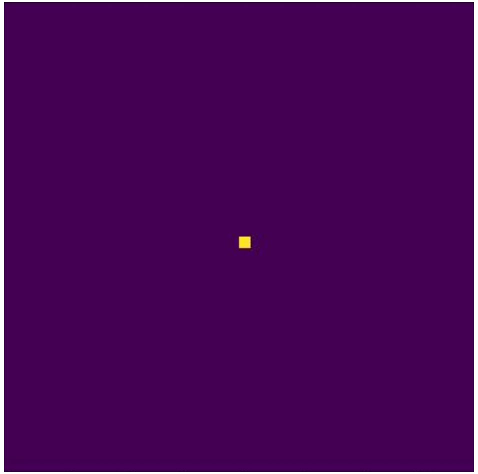
No cell has priority

$$p(i,k \rightarrow i',k') = \frac{(\phi_{i',k'})^{\eta}}{\sum(\phi_{i',k'})^{\eta}}$$



1

0.5



0

2

How DBM generalizes to other models

eta = 1: Diffusion limited aggregation

Linearly proportional to potential strength

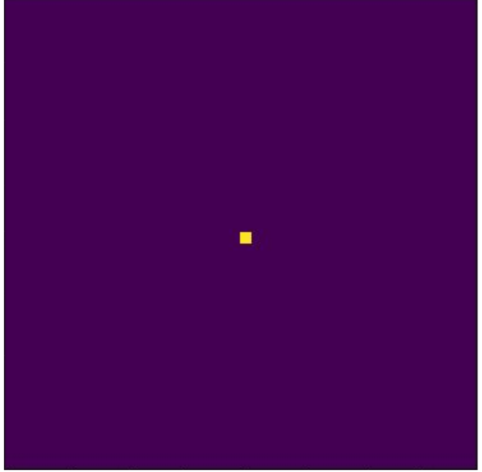
eta = 0: Eden model of cancer growth

No cell has priority

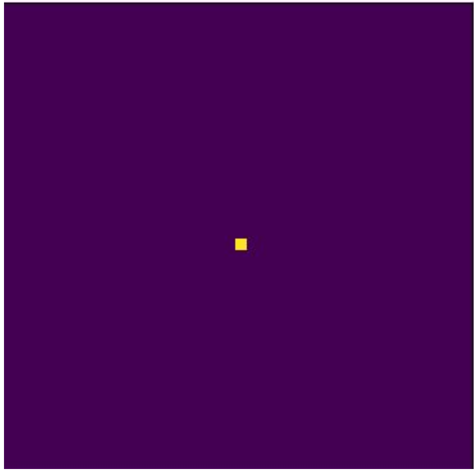
eta = 0.5: Untitled

Cells in relatively strong potential fields have their probabilities more in-line with those in weak fields

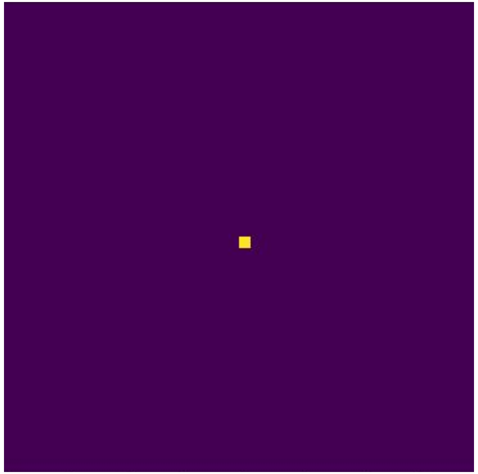
$$p(i,k \rightarrow i',k') = \frac{(\phi_{i',k'})^{\eta}}{\sum(\phi_{i',k'})^{\eta}}$$



1



0.5



0

2

How DBM generalizes to other models

eta = 1: Diffusion limited aggregation

Linearly proportional to potential strength

eta = 0: Eden model of cancer growth

No cell has priority

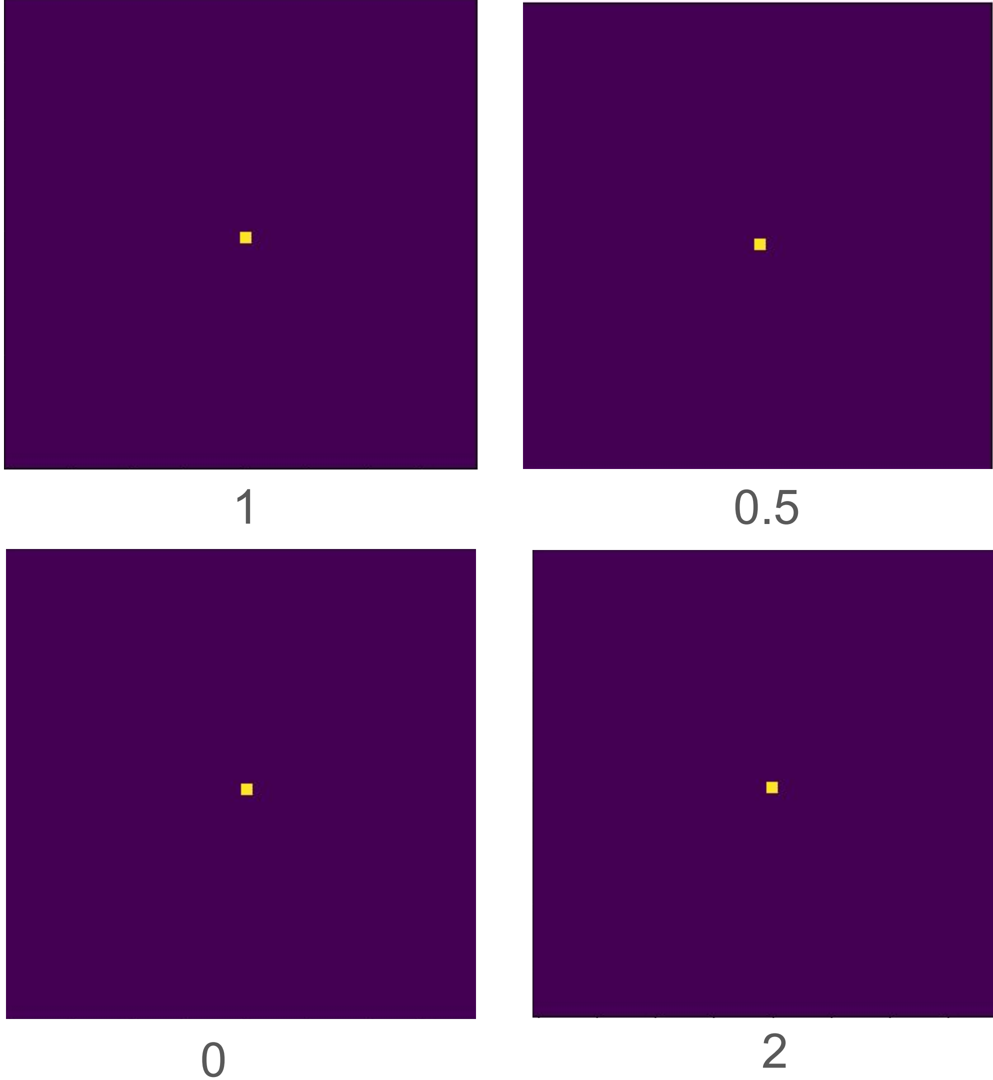
eta = 0.5: Untitled

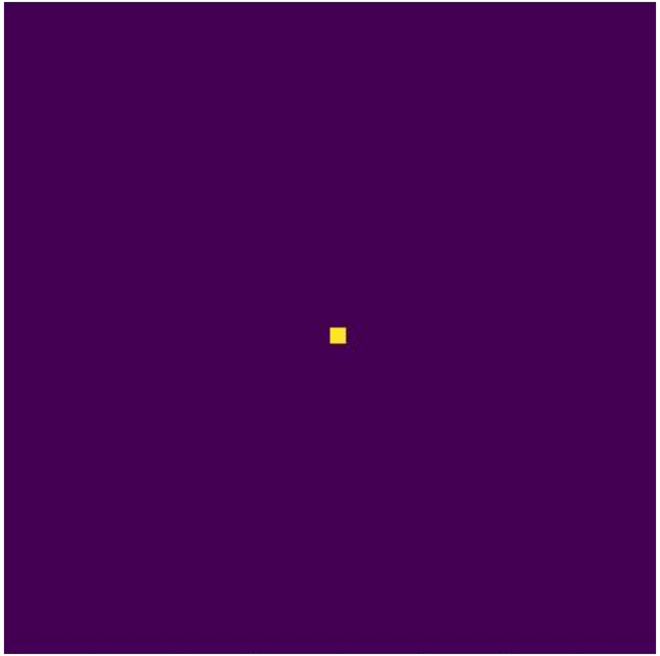
Cells in relatively strong potential fields have their probabilities more in-line with those in weak fields

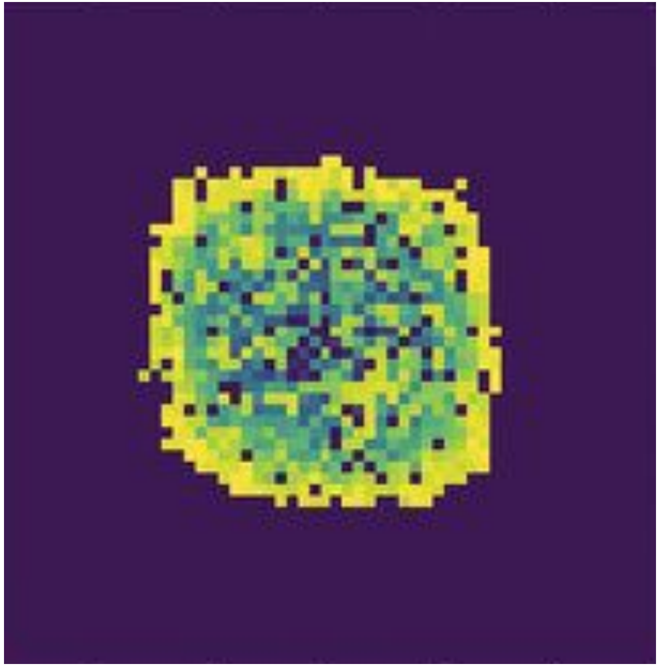
eta = 2:

Cells in relatively strong potential fields have much greater probabilities than those in weak fields

$$p(i,k \to i',k') = \frac{(\phi_{i',k'})^{\eta}}{\sum(\phi_{i',k'})^{\eta}}$$







How DBM generalizes to other models

eta = 1: Diffusion limited aggregation

Linearly proportional to potential strength

eta = 0: Eden model of cancer growth

No cell has priority

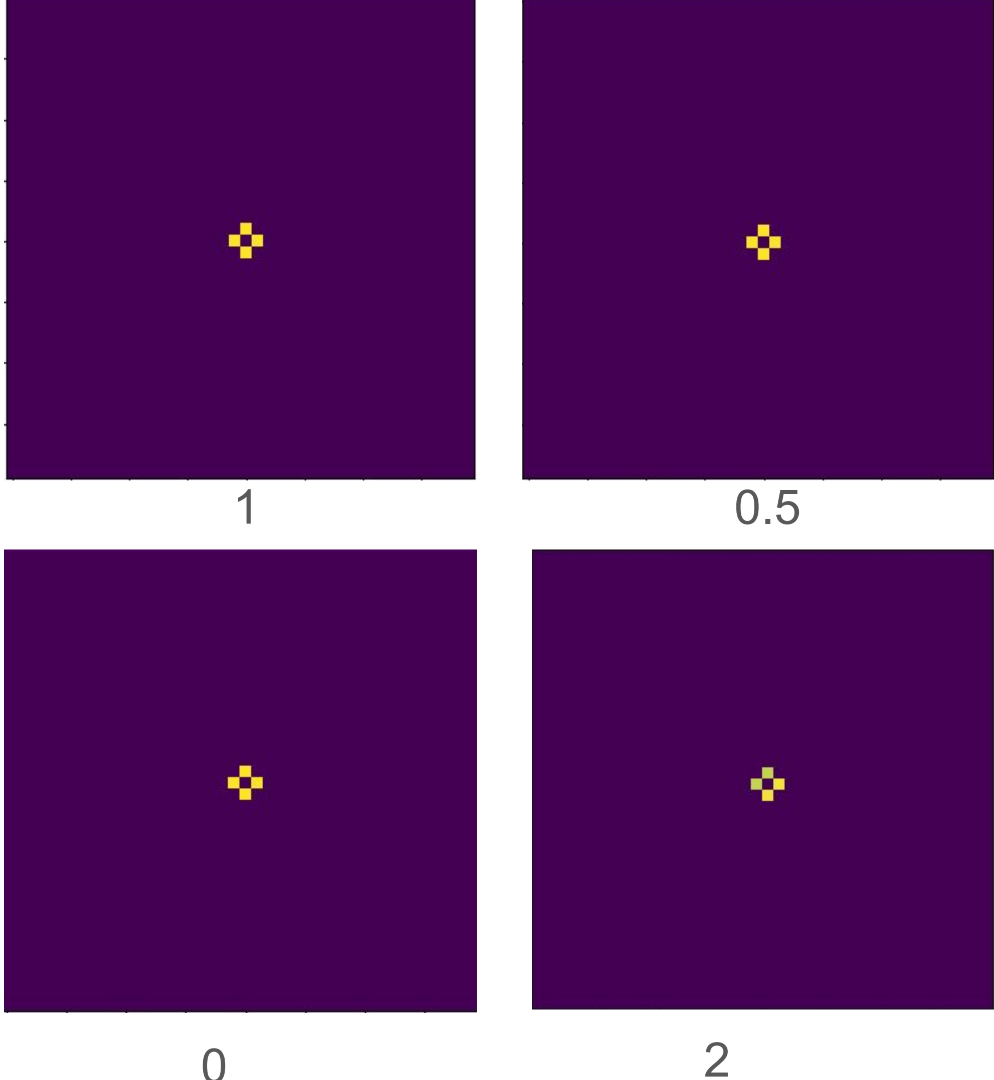
eta = 0.5: Untitled

Cells in relatively strong potential fields have their probabilities more in-line with those in weak fields

eta = 2:

Cells in relatively strong potential fields have much greater probabilities than those in weak fields

$$p(i,k \to i',k') = \frac{(\phi_{i',k'})^{\eta}}{\sum(\phi_{i',k'})^{\eta}}$$



Fractal Dimension of Dielectric Breakdown

L. Niemeyer, L. Pietronero,[a] and H. J. Wiesmann
Brown Boveri Research Center, CH-5405 Baden, Switzerland
(Received 23 November 1983)

It is shown that the simplest nontrivial stochastic model for dielectric breakdown naturally leads to fractal structures for the discharge pattern. Planar discharges are studied in detail and the results are compared with properly designed experiments.

PACS numbers: 77.50.+p, 02.50.+s

Thank you