Surface Growth with Random Tetris Pieces

Release 1.0

Le Chen, Mauricio Montes, Ian Ruau

Dec 03, 2023

CONTENTS

Authors

Le Chen - Affiliation: Department of Mathematics and Statistics, Auburn University - Emails: le.chen@auburn.edu, chenle02@gmail.com

Mauricio Montes - Affiliation: Department of Mathematics and Statistics, Auburn University - Email: mauricio.montes@auburn.edu

Ian Ruau - Affiliation: Department of Mathematics and Statistics, Auburn University - Email: ian.ruau@auburn.edu

CHAPTER

ONE

INTRODUCTION

This Python package offers a suite of tools designed for simulating various random surface growth models. The foundational concepts and methodologies are significantly influenced by the work presented in the book by Barabási and Stanley [BarabasiS95] (1995) on fractal concepts in surface growth.

It is known that the fluctuations in the surface growth models fall into a few universality classes. The most famous one is the Kardar-Parisi-Zhang (KPZ) equation Kardar *et al.* [KPZ86]

$$\frac{\partial h(t,x)}{\partial t} = \nu \nabla^2 h(t,x) + \frac{\lambda}{2}(\nabla h(t,x))^2 + \eta(t,x), \quad t > 0, x \in \mathbb{R}, \tag{1.1}$$

where η is a centered Gaussian noise, which is white in both space and time. The KPZ equation (1.1) is a *stochastic partial differential equation (SPDE)* [Wal86].

KPZ equation (1.1) was solved by Hairer [Hai13] in 2013. The fluctuations of the KPZ equation (1.1) are in the KPZ universality class; see Amir *et al.* [ACQ11]. Numerous discrete models report the same universality class as the KPZ universality class, including

- the ballistic documentation model: [FV85], [MRSB86], [FV85].
- the Eden model: [BKR+88], [PRacz85], [JB85] [MJB86], [ZS86].
- solid-on-Solid model: ...

Theoretical results include the fluctuations of:

- the longest increasing subsequence of a random permutation: [BDJ99].
- the largest eigenvalue of a random matrix: [TW93] and [TW94].
- the asymmetric simple exclusion process (*ASEP*): [Joh00] and [TW09].

CHAPTER

TWO

HISTORY OF THE PACKAGE

The foundational elements of this package, which include the random deposition, random deposition with surface relaxation, the ballistic deposition models, and a visualization component, were initially crafted by the first author. These components, forming the basis of the package, were part of a graduate student seminar in October 2023, as documented in Chen [Che23a]. The seminar's slides provide an excellent introduction to these foundational aspects of the package.

Building upon this foundation, the package has been enriched with simulations for growth models with Tetris pieces. This feature is to test the belief of the universality of the growth models. This feature was developed as a final course project for the course *"Math 7820: Applied Stochastic Processes I,"* undertaken by the second and third authors in Fall 2023. This is still an ongoing project. More feature and functionalities will be added in the future. Some simulation experiments will be carried out to test the universality of the growth models.

CHAPTER

THREE

ACKNOWLEDGMENTS

The references throughout this document have been meticulously compiled and are available in a comprehensive bibliography bank [Che23b].

This work is partially supported by the **National Science Foundation (NSF)** under Grant No. No. 2246850 (2023–2026) and the collaboration/travel award from the **Simons foundation** under Award No. 959981 (2022–2027).

CHAPTER

FOUR

BIBLIOGRAPHY

CHAPTER

FIVE

RD_CLI MODULE

This code was written initially for the Graduate Student Seminar [Che23a]

RD_CLI.**Ballistic_Deposition**(*width*, *height*, *steps*)

Simulate Ballistic Deposition on a substrate. In this simulation, particles stick upon contact with the substrate or a deposited particle.

This is simulation for snowflakes piling.

Parameters

- **width** (*int*) – Width of the substrate.
- **height** (*int*) – Height of the substrate.
- **steps** (*int*) – Number of particles to drop.

Returns
Filename of the output substrate for Ballistic Deposition; A csv file contains the substrate state.

Return type
string

RD_CLI.**Envelop**(*substrate*)

Compute the top envelope of a substrate.

This function calculates the highest particle position in each column of the substrate. It is used to visualize the top envelope of the substrate in the simulation.

Parameters
substrate (*numpy.ndarray*) – The substrate matrix to compute the envelop for.

Returns
Array representing the top envelope of the substrate.

Return type
numpy.ndarray

RD_CLI.**Random_Deposition**(*width*, *height*, *steps*)

This is a function to simulate Random Deposition on a substrate. This is simulation for the independent boxes (sand) piling. All columns are independent.

Parameters

- **width** (*int*) – Width of the substrate.
- **height** (*int*) – Height of the substrate.
- **steps** (*int*) – Steps or times to run.

Returns
Filename of the output substrate.

Return type
string

RD_CLI.**Random_Deposition_Surface_Relaxation**(*width*, *height*, *steps*)

This is a function to simulate Random Deposition on a substrate with the surface relaxation. Particle will seek the lowest left/right neighbors to land.

This is simulation for the independent boxes (sand) piling with surface relaxation.

Parameters

- **width** (*int*) – Width of the substrate.
- **height** (*int*) – Height of the substrate.
- **steps** (*int*) – Steps or times to run.

Returns
Filename of the output substrate.

Return type
string

RD_CLI.**interface_width**(*filename*)

Compute and visualize the interface width of a substrate from a given simulation.

This function reads the substrate state from a file, calculates the interface width over time, and generates a log-log plot of the interface width. It also computes the slope of the log-log plot as a function of time.

Parameters
filename (*str*) – The name of the file containing the substrate data.

Returns
Array containing the interface width calculated at each step; An image file with the same name as the input file, but with a .png extension. The file contains the statistical figures.

Return type
numpy.ndarray

RD_CLI.**main**()

The main function to simulate different types of surface growth models based on the provided command-line arguments.

This function sets up a command-line interface for simulating Random Deposition, Random Deposition with Surface Relaxation, or Ballistic Decomposition on a substrate. It accepts various parameters like width, height, and number of steps for the simulation. It also provides options for generating a movie of the simulation and calculating interface width.

The function decides the type of simulation based on the arguments passed, performs the simulation, and then proceeds to calculate the interface width. If the movie generation option is selected, it invokes another script to generate the movie.

To use the script from terminal, the following options are expected:

-w, --width : Width of the substrate (default: 100)

-e, --height : Maximum height of the substrate (default: 60)

-s, --steps : Number of particles to drop (default: 5000)

-r, --relax : Enable surface relaxation (default: False)

-b, --BD : Enable ballistic decomposition (default: False)

-m, --movie : Generate an mp4 movie of the simulation (default: False)

It returns:

1. A text file representing the substrate state.
2. Statistical figures, including a log-log plot for the interface width and the estimated slope.
3. (Optional) An mp4 movie of the simulation process.

Example:

```
> ptyhon3 RD_CLI.py -w 100 -e 60 -s 5000 --BD --movie
```

In this example, the script will simulate Ballistic Decomposition on a substrate of size 100x60 for 5000 steps. And the simulation movie will be generated.

CHAPTER

SIX

VISUALIZE_RD MODULE

Visualize_RD.**main**()

Main function to execute the particle deposition visualization script.

This script visualizes the process of particle deposition based on the data from a substrate simulation. It generates a video showing the evolution of the deposition over time. The script offers several command-line options to customize the visualization, such as showing the top envelope, the average height, and adjusting the video playback rate.

Options:

-f, --file : Required. Path to the substrate file containing simulation data.

-t, --title : Optional. Title of the plot. If not provided, no title is displayed.

-r, --rate : Optional. Frame rate for the video. Default is 4 frames per second.

-e, --envelop : Optional. Flag to display the top envelope of the deposition. Default is False.

-a, --average : Optional. Flag to display the average height of the deposition. Default is False.

-p, --play : Optional. Flag to automatically play the video after generation. Default is False.

Example:

```
> python3 Visualize_RD.py -f substrate.txt -t "Simulation Title" -r 10 --envelop --
↪average --play
```

This command will run the script on 'substrate.txt', set a title, use a frame rate of 10 fps, and display both the top envelope and average height in the visualization.

Visualize_RD.**visualize_simulation**(*filename*, *plot_title*, *rate*, *envelop*, *show_average*)

Visualize the particle deposition simulation and generate a video.

This function loads the substrate data from a file and visualizes the deposition process as an animation. It supports visualizing the top envelope and average height of the deposited particles. The final output is saved as an mp4 video file.

Parameters

- **filename** (*str*) – The path to the substrate data file.
- **plot_title** (*str*) – The title of the plot.
- **rate** (*int*) – The frame rate for the video.
- **envelop** (*bool*) – Flag to indicate whether to show the top envelope.

- **show_average** (*bool*) – Flag to indicate whether to show the average height.

Returns
The filename of the generated mp4 video.

Return type
str

CHAPTER

SEVEN

TETRIS_COMPLETE MODULE

This module simulates the surface growth by Tetris pieces. It includes functions to generate random Tetris pieces, calculate their landing positions on a substrate, and simulate a game of Tetris for a given number of steps and a defined grid size.

By Ian Ruau (iir0001@auburn.edu) and Mauricio Montes (mauricio.montes@auburn.edu) Date: 12/2023

tetris_complete.**Tetris_Choice**()

Randomly selects a Tetris piece and its orientation.

There are 7 Tetris pieces:

- 0 : the square;
- 1 : the line;
- 2 : the L;
- 3 : the J;
- 4 : the T;
- 5 : the S;
- 6 : the Z.

There are 4 orientations for each piece:

- 0 is the original orientation;
- 1 is the 90 degree rotation;
- 2 is the 180 degree rotation;
- 3 is the 270 degree rotation.

Returns

A 2-element array: the first element is the piece type (0-6); the second element is the orientation (0-3).

Return type

numpy.ndarray

To-Do's

- Add input file to specify the probability of each piece.

tetris_complete.**Tetris_RD**(*width*, *height*, *steps*)

This function simulates the Tetris Decomposition model on a substrate.

Parameters

- **width** (*int*) – The width of the substrate.
- **height** (*int*) – The height of the matrix.
- **steps** (*int*) – The steps to simulate.

Returns
Filename of the output file.

Return type
string

tetris_complete.**ffnz**(*matrix*, *height*, *column*)

Finds the first non-zero entry in a specified column of a matrix.

Parameters

- **matrix** (*numpy.ndarray*) – The matrix to search.
- **height** (*int*) – The height of the matrix.
- **column** (*int*) – The column index to search in.

Returns
The index of the first non-zero entry.

Return type
int

tetris_complete.**main**()

To use the script from terminal, the following options are expected:

-w, --width : Width of the substrate (default: 100)

-e, --height : Maximum height of the substrate (default: 60)

-s, --steps : Number of particles to drop (default: 5000)

It returns:

1. A text file representing the substrate state.

Example

```
> ptyhon3 tetris_complete.py -w 100 -e 60 -s 5000
```

In this example, the script will simulate Tetris Decomposition on a substrate of size 100x60 for 5000 steps.

CHAPTER

EIGHT

INDICES AND TABLES

- genindex
- modindex
- search

BIBLIOGRAPHY

[ACQ11] Gideon Amir, Ivan Corwin, and Jeremy Quastel. Probability distribution of the free energy of the continuum directed random polymer in $1+1$ dimensions. *Comm. Pure Appl. Math.*, 64(4):466–537, 2011. URL: https://doi.org/10.1002/cpa.20347, doi:10.1002/cpa.20347.

[BDJ99] Jinho Baik, Percy Deift, and Kurt Johansson. On the distribution of the length of the longest increasing subsequence of random permutations. *J. Amer. Math. Soc.*, 12(4):1119–1178, 1999. URL: https://doi.org/10.1090/S0894-0347-99-00307-0, doi:10.1090/S0894-0347-99-00307-0.

[BKR+88] R. Baiod, D. Kessler, P. Ramanlal, L. Sander, and R. Savit. Dynamical scaling of the surface of finite-density ballistic aggregation. *Phys. Rev. A*, 38:3672–3679, Oct 1988. URL: https://link.aps.org/doi/10.1103/PhysRevA.38.3672, doi:10.1103/PhysRevA.38.3672.

[BarabasiS95] Albert-László Barabási and H. Eugene Stanley. *Fractal concepts in surface growth*. Cambridge University Press, Cambridge, 1995. ISBN 0-521-48318-2. URL: https://doi.org/10.1017/CBO9780511599798, doi:10.1017/CBO9780511599798.

[Che23a] Le Chen. Graduate student seminars by Le Chen. nov 2023. URL: https://doi.org/10.5281/zenodo.10206966, doi:10.5281/zenodo.10206966.

[Che23b] Le Chen. SPDEs-Bib: a comprehensive bibliography of stochastic partial differential equations and related topics. nov 2023. URL: https://doi.org/10.5281/zenodo.10143431, doi:10.5281/zenodo.10143431.

[FV85] Fereydoon Family and Tamás Vicsek. Scaling of the active zone in the eden process on percolation networks and the ballistic deposition model. *Journal of Physics A: Mathematical and General*, 18(2):L75, feb 1985. URL: https://dx.doi.org/10.1088/0305-4470/18/2/005, doi:10.1088/0305-4470/18/2/005.

[Hai13] Martin Hairer. Solving the KPZ equation. *Ann. of Math. (2)*, 178(2):559–664, 2013. URL: https://doi.org/10.4007/annals.2013.178.2.4, doi:10.4007/annals.2013.178.2.4.

[Joh00] Kurt Johansson. Shape fluctuations and random matrices. *Comm. Math. Phys.*, 209(2):437–476, 2000. URL: https://doi.org/10.1007/s002200050027, doi:10.1007/s002200050027.

[JB85] R Jullien and R Botet. Scaling properties of the surface of the eden model in d=2, 3, 4. *Journal of Physics A: Mathematical and General*, 18(12):2279, aug 1985. URL: https://dx.doi.org/10.1088/0305-4470/18/12/026, doi:10.1088/0305-4470/18/12/026.

[KPZ86] Mehran Kardar, Giorgio Parisi, and Yi-Cheng Zhang. Dynamic scaling of growing interfaces. *Phys. Rev. Lett.*, 56(9):889, 1986. URL: https://doi.org/10.1103/PhysRevLett.56.889, doi:10.1103/PhysRevLett.56.889.

[MJB86] P. Meakin, R. Jullien, and R. Botet. Large-scale numerical investigation of the surface of eden clusters. *Europhysics Letters*, 1(12):609, jun 1986. URL: https://dx.doi.org/10.1209/0295-5075/1/12/001, doi:10.1209/0295-5075/1/12/001.

[MRSB86] Paul Meakin, P. Ramanlal, L. M. Sander, and R. C. Ball. Ballistic deposition on surfaces. *Phys. Rev. A*, 34:5091–5103, Dec 1986. URL: https://link.aps.org/doi/10.1103/PhysRevA.34.5091, doi:10.1103/PhysRevA.34.5091.

[PRacz85] Michael Plischke and Zoltán Rácz. Dynamic scaling and the surface structure of eden clusters. *Phys. Rev. A*, 32:3825–3828, Dec 1985. URL: https://link.aps.org/doi/10.1103/PhysRevA.32.3825, doi:10.1103/PhysRevA.32.3825.

[TW93] Craig A. Tracy and Harold Widom. Level-spacing distributions and the Airy kernel. *Phys. Lett. B*, 305(1-2):115–118, 1993. URL: https://doi.org/10.1016/0370-2693(93)91114-3, doi:10.1016/0370-2693(93)91114-3.

[TW94] Craig A. Tracy and Harold Widom. Level spacing distributions and the Bessel kernel. *Comm. Math. Phys.*, 161(2):289–309, 1994. URL: http://projecteuclid.org/euclid.cmp/1104269903.

[TW09] Craig A. Tracy and Harold Widom. Asymptotics in ASEP with step initial condition. *Comm. Math. Phys.*, 290(1):129–154, 2009. URL: https://doi.org/10.1007/s00220-009-0761-0, doi:10.1007/s00220-009-0761-0.

[Wal86] John B. Walsh. An introduction to stochastic partial differential equations. In *École d'été de probabilités de Saint-Flour, XIV—1984*, volume 1180 of Lecture Notes in Math., pages 265–439. Springer, Berlin, 1986. URL: https://doi.org/10.1007/BFb0074920, doi:10.1007/BFb0074920.

[ZS86] J. G. Zabolitzky and D. Stauffer. Simulation of large eden clusters. *Phys. Rev. A*, 34:1523–1530, Aug 1986. URL: https://link.aps.org/doi/10.1103/PhysRevA.34.1523, doi:10.1103/PhysRevA.34.1523.

PYTHON MODULE INDEX

INDEX