Scientific Computing II

The Return of the SciCoPyth

Jaro Camphuijsen (6042473) and Rahiel Kasim (10447539)

May 12, 2016

Contents

1 Diffusion Limited Aggregation

Diffusion Limited Aggregation (DLA) considers diffusing particles that that can stick to a certain structure and thereby expand the structure. Because we are dealing with diffusion we should solve the diffusion equation at each growth step to calculate the new concentration of particles.

$$\frac{\partial \phi(r,t)}{\partial t} = D\nabla^2 \phi(r,t) \tag{1}$$

Because the growth of such a structure (e.g. a coral or mineral structure) is on a very long timescale and the structure growth per step is very small we can assume the diffusion reaches equilibrium before the next step and we can solve the time independent diffusion equation.

The update scheme of a growth simulation of a certain number of `steps` is given in the following pseudo code snippet:

```
def grow(eta, omega, steps):
    for n in range(steps):
        nutLattice = SOR(omega, objLattice)
        objLattice = grow_step(eta, nutLattice)
```

Here we have the nutrient lattice `nutLattice` which is updated each time step by the `SOR` function and contains the nutrient concentration at each cell, it uses the object lattice `objLattice` for its boundary conditions. The object lattice is updated using the `grow_step` function to calculate the probability for

each cell to become part of the structure from the nutrient concentration in the cell.

To simulate the growth we define two different cell states: a cell can be part of the structure or it can be empty. Each step we determine all grow candidates. For a cell to be a grow candidate it must be empty and has one of his neighbours be part of the structure. Each candidate has the following probability to switch it state from empty to be part of the structure:

$$p_g(i,j) = \frac{c_{i,j}^{\eta}}{\sum_{\text{candidates}} c_{i,j}^{\eta}} \tag{2}$$

For the calculation of the time independent diffusion equation we use the successive over-relaxation (SOR) function from the last assignment. In the previous assignment [1] we also optimized the relaxation parameter ω, however the optimal value of ω depends on the boundary values of the domain. These were fixed in the simple example of the previous exercise but since we have a growing object this time, the value of ω should in principle be optimized every time the structure has grown a bit. This to computationally demanding to be of any use.

When we do a simulation with a growing parameter $0 < \eta < 1$ we get a problem using the SOR calculation. Due to the over relaxation it is possible to end up with a small negative concentration. This is no problem for other values of η, however since the root of a negative number is not defined in our program we end up with an error. To avoid this problem in calculation and because we can not truly optimize ω which would result in a minor speed improvement anyway, we should set $\omega = 1$ or use under relaxation for all simulations with $0 < \eta < 1$.

For varying growing parameter η we get different structures as shown in figure 1. If we set η to zero as in figure 1a we turn off the influence of the nutrient concentration. Each grow candidate has the same probability of growing which results in a circular bulk around the initial point. With increasing η the bulk tends to grow towards more expanded structures with normal DLA growth for $\eta = 1$ in figure 1c. If we keep increasing η the influence of the nutrient concentration increases and since we have a gradient directed to the top of the domain, the structure will grow more linearly with less pronounced branches. In the case of $\eta = 3$ as seen in figure 1f, the structure is almost completely linear this is because candidate cells at the top of the structure have a much higher probability to become part of the object due to the concentration gradient.

2 Monte Carlo DLA

A different method to simulate DLA uses random walkers and therefore is a Monte Carlo type simulation. This type of simulation can in principle be seen as a true imitation of diffusion since diffusing particles in the real world also follow a random walk. For this simulation we created a class `Walker` with methods to take a random step, check its neighbours and merge with the structure in the object lattice whenever it touches the object. Additionally a sticking parameter (p_{stick}) can be implemented which determines the probability that the walker merges if it touches the structure. If this probability is one we are left with the original Monte Carlo simulation. The result for different values of this

sticking probability can be seen in figure 2. We can clearly see the similarity between Monte-Carlo DLA with $p_{stick} = 1$ in figure 2a and the normal DLA simulation in figure 1c. Also we see that for lower values of p_{stick} we get more linear structures with less and smaller branches. This is equivalent to raising the growing parameter η in the normal DLA simulation.

The advantage of the Monte Carlo method is that we do not have to solve the diffusion equations which is the bottle neck in the DLA simulation, however doing a pure random until the particle sticks takes a long time. We can improve on the runtime dramatically by adding a small bias for steps downward as is done in some simulations. [2] However this breaks the similarity with true diffusion since we introduce a net flow in the system. Of course we can use this flow as it is present in several natural systems (e.g. gravity pulling on the nutrients in the formation of corals).

3 The Gray-Scott Model

The Gray-Scott Model is a reaction-diffusion system. It models the flow of two chemicals, u and v, by showing their concentrations over time as they diffuse and react with each other. Reaction-diffusion systems look like the familiar diffusion equation but they have an extra term that is a function of the relevant concentrations (e.g. $f(u, v)$). The Gray-Scott model can be expressed as:

$$\begin{aligned} \frac{\partial u}{\partial t} &= D_u \nabla^2 u - uv^2 + f(1-u), \\ \frac{\partial v}{\partial t} &= D_v \nabla^2 v + vv^2 - (f+k)v, \end{aligned} \tag{3}$$

where D_u and D_v are the diffusion constants of the chemicals and f and k parameters of the underlying chemical reaction. To solve these partial differential equations for u and v we discretize the equations using finite difference methods. We divide 2D space in N chunks of δx, δy and time in periods of δt. In particular, we substitute

$$\begin{aligned} \frac{\partial u}{\partial t} &= \frac{u_i^{k+1} - u_i^k}{\delta t}, \\ \nabla^2 u &= \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right)^2 u, \\ &= \frac{1}{\delta x^2}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}), \end{aligned}$$

in system 3 to get the discretization:

$$\begin{aligned} u_{i,j}^{k+1} &= u_{i,j}^k + \delta t \left(\frac{D_u}{\delta x^2}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}) - uv^2 + f(1-u) \right) \\ v_{i,j}^{k+1} &= v_{i,j}^k + \delta t \left(\frac{D_v}{\delta x^2}(v_{i+1,j} + v_{i-1,j} + v_{i,j+1} + v_{i,j-1} - 4v_{i,j}) + uv^2 - (f+k)v \right) \end{aligned}$$

We take periodic boundary conditions along the x and y directions: $u(0, y, t) = u(N\delta x, y, t)$ and $u(x, 0, t) = u(x, N\delta y, t)$, with the same for v. We've implemented the derived discretization with these boundary conditions.

For the parameters we've chosen $\delta t = \delta x = \delta y = 1$, $N = 300$, $D_u = 0.16$, $D_v 0.08$, $f = 0.035$ and $k = 0.060$. Our initial conditions were $u = 0.5$ everywhere and $v = 0.25$ in a square around the origin and $v = 0$ elsewhere. We added a small amount of noise to the initial conditions of u and v of magnitude 0.05.

In figure 3 we see the concentrations of u and v over time. First at $t = 0$ we see the square of the initial conditions with the added noise seen as speckles all over the domain. In the middle at $t = 700$ we see that the noise has disappeared due to diffusion. The square itself has also disappeared, leaving behind four circles growing at its former corners. And at the bottom at $t = 1300$ the circles have grown into four star-like shapes.

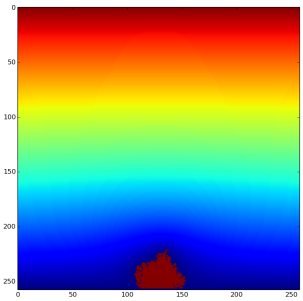
The process continues in figure 4. The four stars have merged to a nice symmetric emblem. In the middle we see some breaking of symmetry, this could indicate the effect of numerical error, as the initial conditions were isotropic. Finally at the bottom the pattern has grown to extend over the boundaries. Because we implemented periodic boundary conditions, this final pattern can be used on tiles with the edges fitting perfectly.

We now explore the patterns formed when the parameters are changed slightly: $f = 0.030$ and $k = 0.055$. Everything else is the same. This time we omit u from the plots, because it does not add new information on the patterns given v. In figure 5 we see the formation of the patterns. The initial square grows into a waffle (b and c) and continues expanding until we end up with many small mountains (f), or spots. We conclude by noting that the small change in parameters lead to quite different patterns.

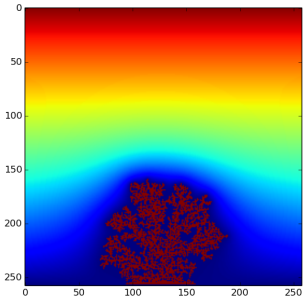
References

[1] J. Camphuijsen, R. Kasim. Scientific Computing 1 - Exploring the SciCoPyth's mind. 2016.

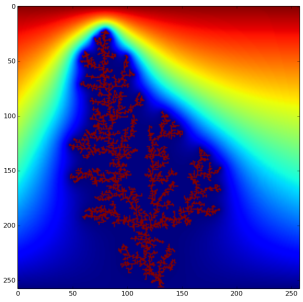
[2] Jason Davies. Diffusion-limited aggregation. [`https://www.jasondavies.com/dla/`; accessed 10-May-2016].



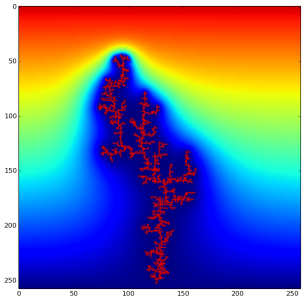
(a) $\eta = 0$



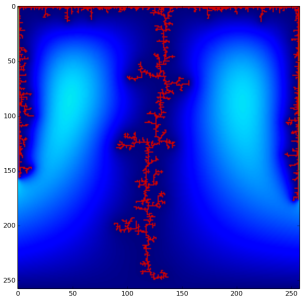
(b) $\eta = 0.5$



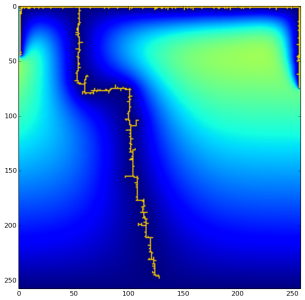
(c) $\eta = 1$



(d) $\eta = 1.5$

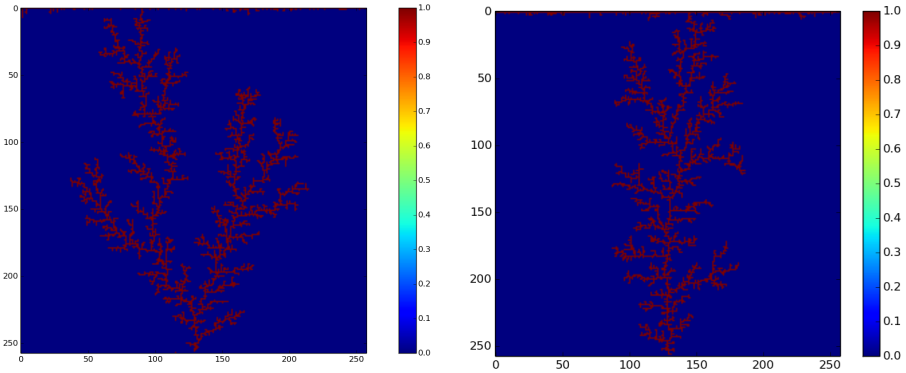


(e) $\eta = 2$

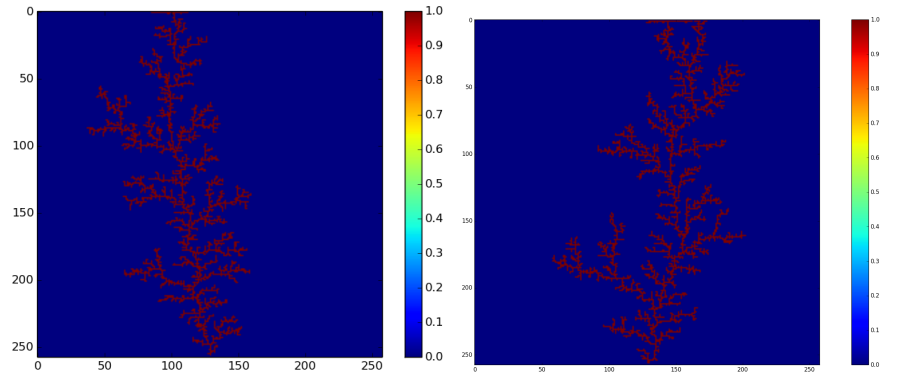


(f) $\eta = 3$

Figure 1: Diffusion Limited Aggregation for different values of η



(a) $p_{stick} = 1$ (b) $p_{stick} = 0.5$



(c) $p_{stick} = 0.2$ (d) $p_{stick} = 0.1$

Figure 2: A Monte Carlo simulation of Diffusion Limited Aggregation for different values of the sticking probability p_{stick}. These structures can be compared to the structures in figure 1c-1e

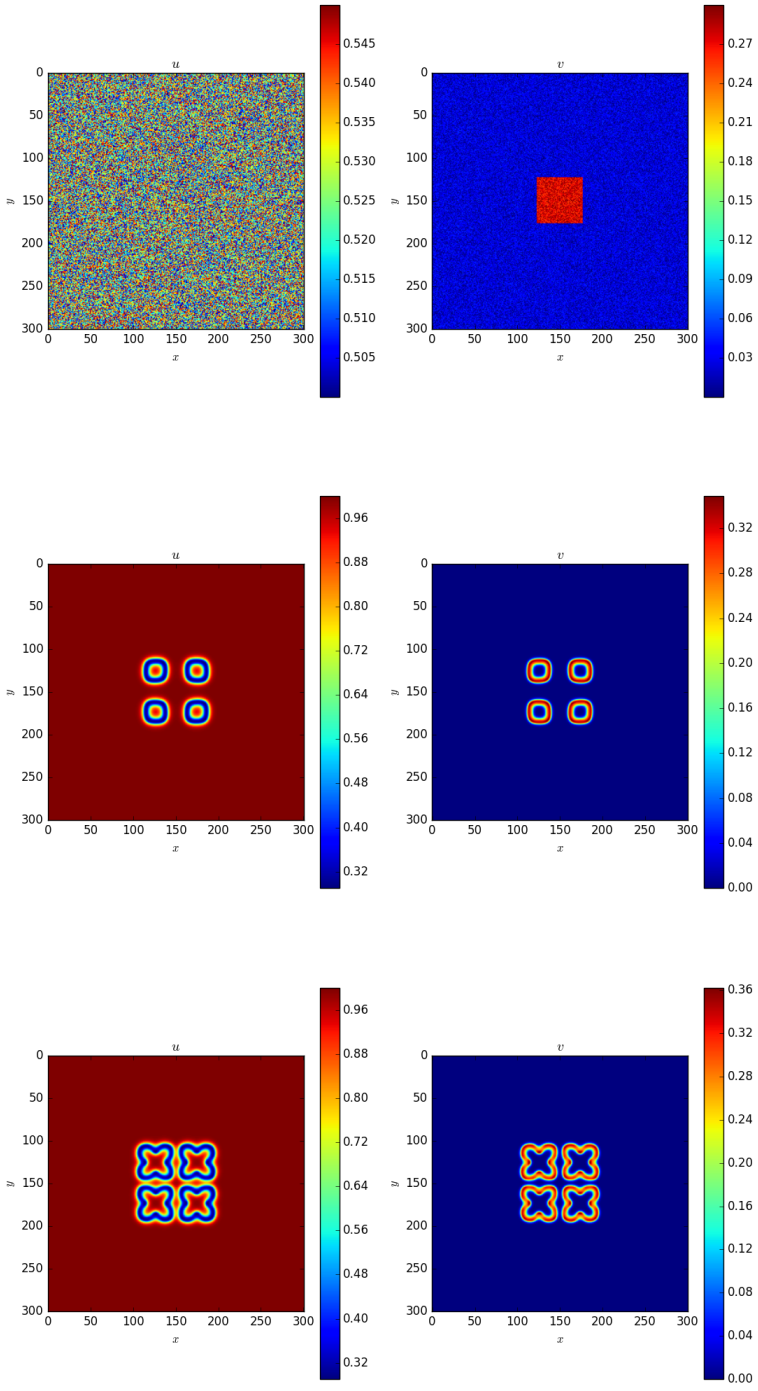


Figure 3: The patterns that arise in the Gray-Scott model with the parameters described in the text. At the top we see the concentrations at time $t = 0$, in the middle at $t = 700$ and at the bottom at $t = 1300$.

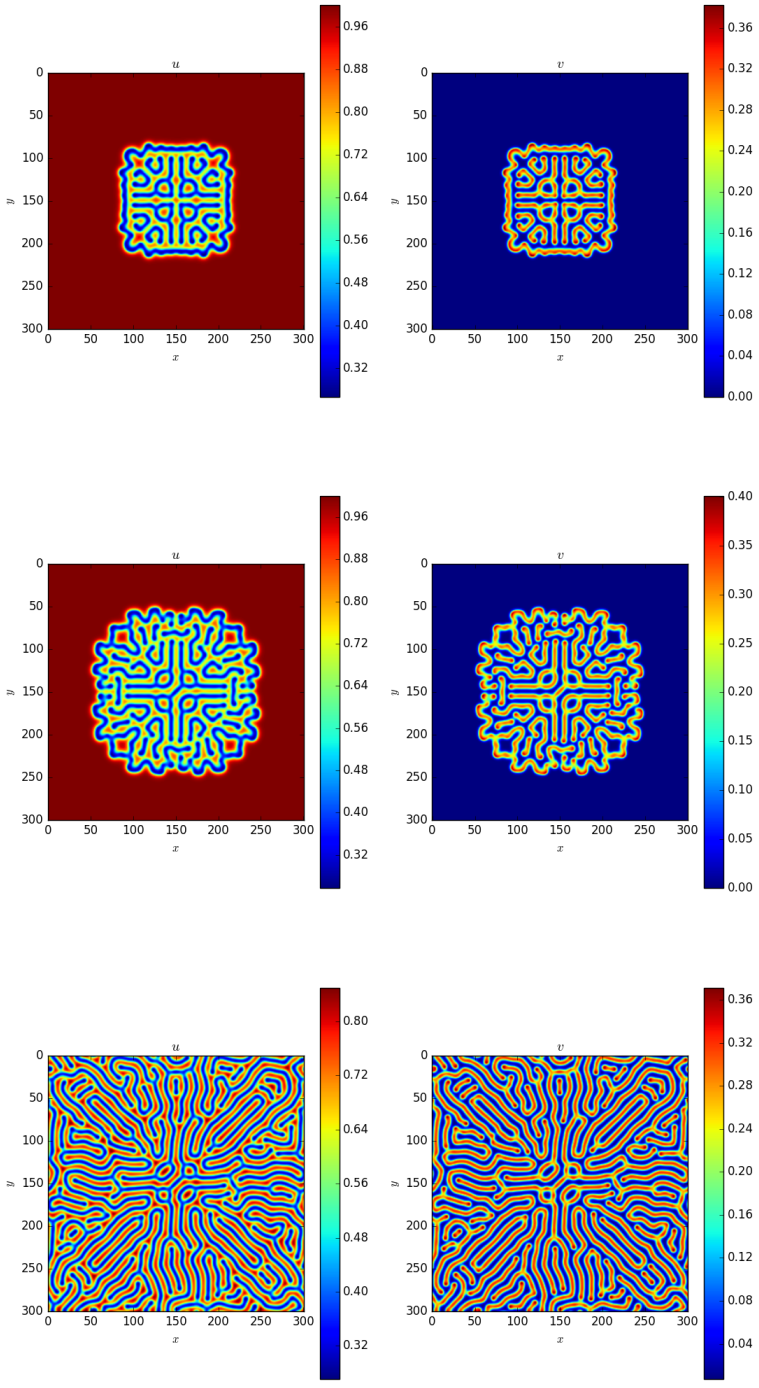
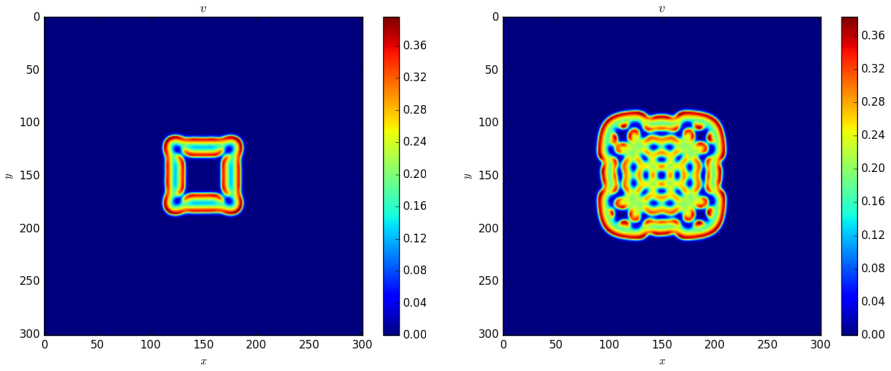
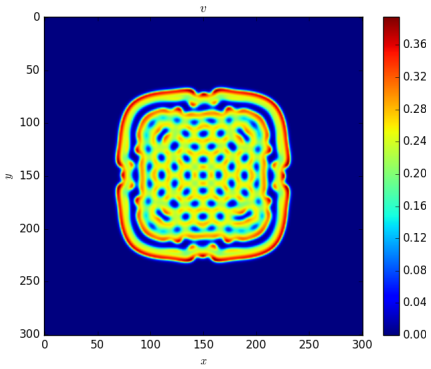


Figure 4: The continuation of figure 3. Top: $t = 2500$, middle: $t = 4000$, bottom: $t = 10000$.

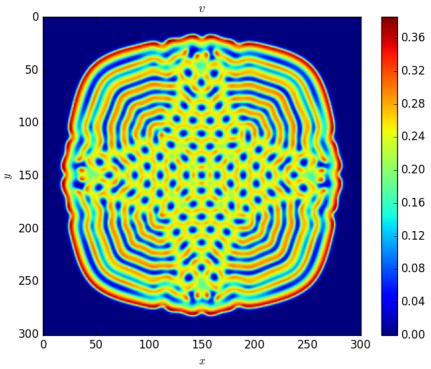


(a) $t = 350$

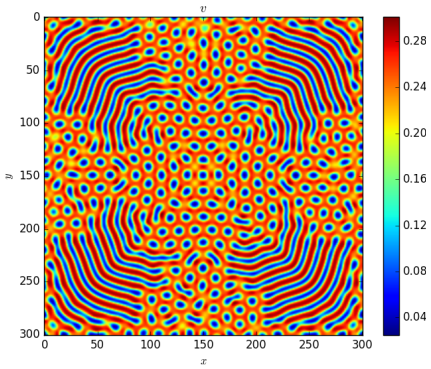
(b) $t = 1000$



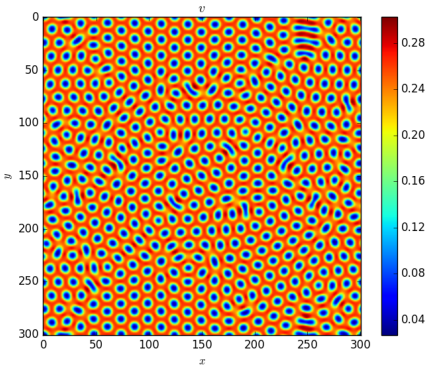
(c) $t = 1600$



(d) $t = 3000$



(e) $t = 5000$



(f) $t = 50000$

Figure 5: The Gray-Scott model with $f = 0.030$ and $k = 0.055$.