AM225: Assignment 1 solutions*

Problem 1 – A randomized Casino game

Part (a)

The expected amount of winnings is computed to be \$271.83. Table 1 shows the wall clock times to simulate 10^9 instances of the casino game using 1, 2, and 4 threads. The parallel efficiency is

Threads	Time	Efficiency†
1	24.8 s	1.00
2	12.3 s	1.01
4	6.3 s	0.98

Table 1: Wall clock times for the casino game.

near-perfect for this problem. Once the each thread is launched, it can compute its random trials without requiring any coordination with other threads.

Part (b)

Let N be a random variable denoting the number of uniformly distributed random samples needed for the sum to exceed 1. Since N is realized as a natural number, the expected value of N is given by

$$\begin{aligned}
\mathbb{E}[N] &= \sum_{n=0}^{\infty} n\mathbb{P}(N = n) \\
&= \sum_{n=0}^{\infty} n\left(\mathbb{P}(N > n-1) - \mathbb{P}(N > n)\right) \\
&= \sum_{n=1}^{\infty} n\mathbb{P}(N > n-1) - \sum_{n=1}^{\infty}(n-1)\mathbb{P}(N > n-1) \\
&= \sum_{n=0}^{\infty} \mathbb{P}(N > n).
\end{aligned}$$

$\mathbb{P}(N > n)$ is the probability that the sum of the first n samples is less than (or equal to) 1, which is equal to the volume of the standard n-simplex $\{x \in \mathbb{R}^n_{\geq 0} \mid x_1 + x_2 + \cdots + x_n \leq 1\}$, which is $1/n!$. Thus,

$$\mathbb{E}[N] = \sum_{n=0}^{\infty} \mathbb{P}(N > n) = \sum_{n=0}^{\infty} \frac{1}{n!} = e \approx 2.7183.$$

Since we receive \$100 for each number drawn, the expected winnings are \$271.83. As it costs us only \$250 to play, the expected profit is \$21.83.

*Solutions to problems 2, 3, 5, and 6 written by Nick Derr. Solutions to problems 1 and 4 written by Dan Fortunato.

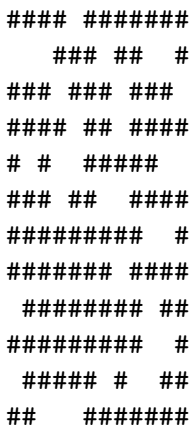
†Efficiency is computed as (time using one thread)/($n \times$ time using n threads).

Problem 2 – Cellular automaton mazes

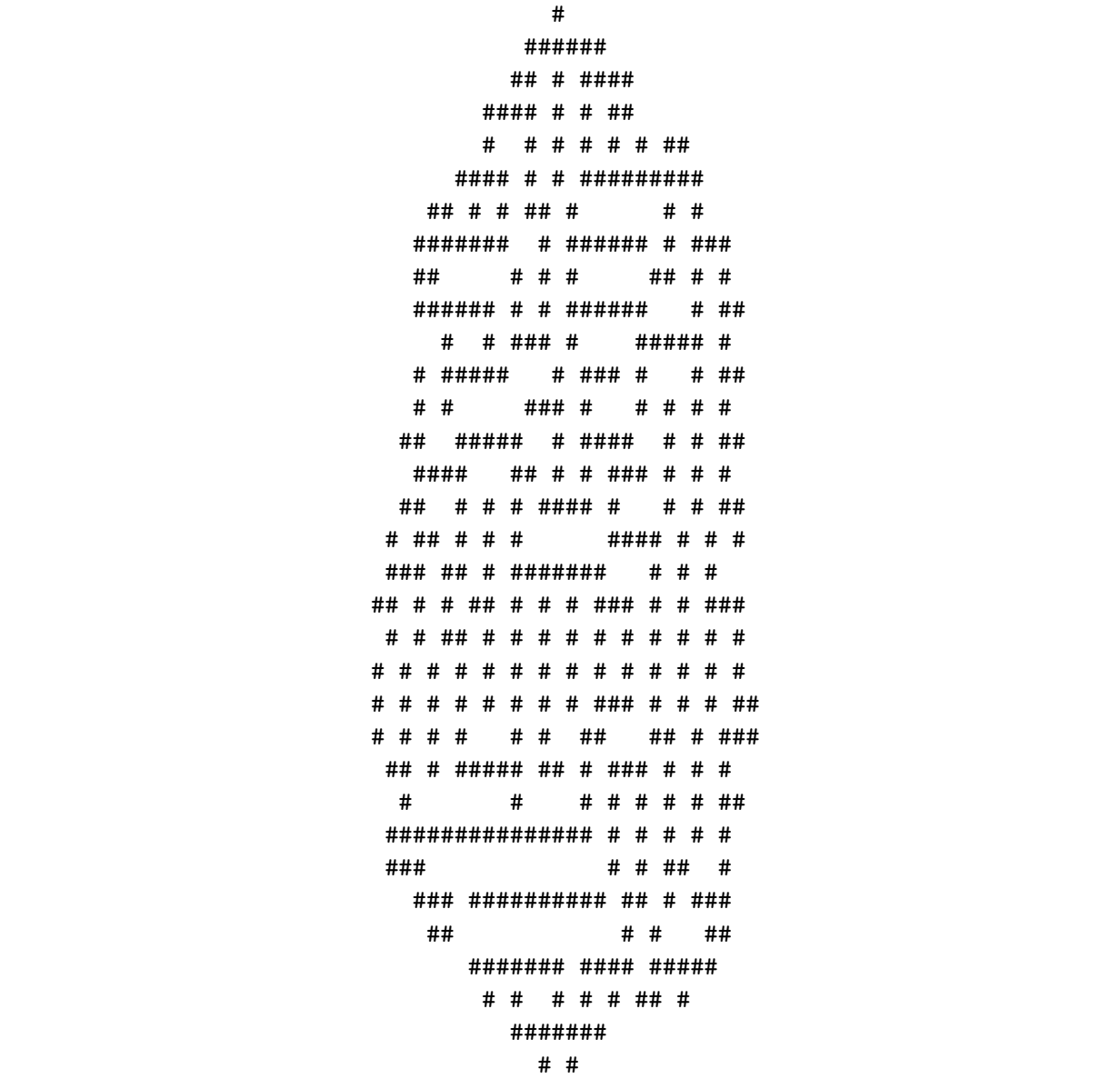
Part (a)

Characteristic snapshots of the initial condition and subsequent six generations are as follows:

```
generation 0:
--------------------------------------------------------------------------------
```



generation 1:



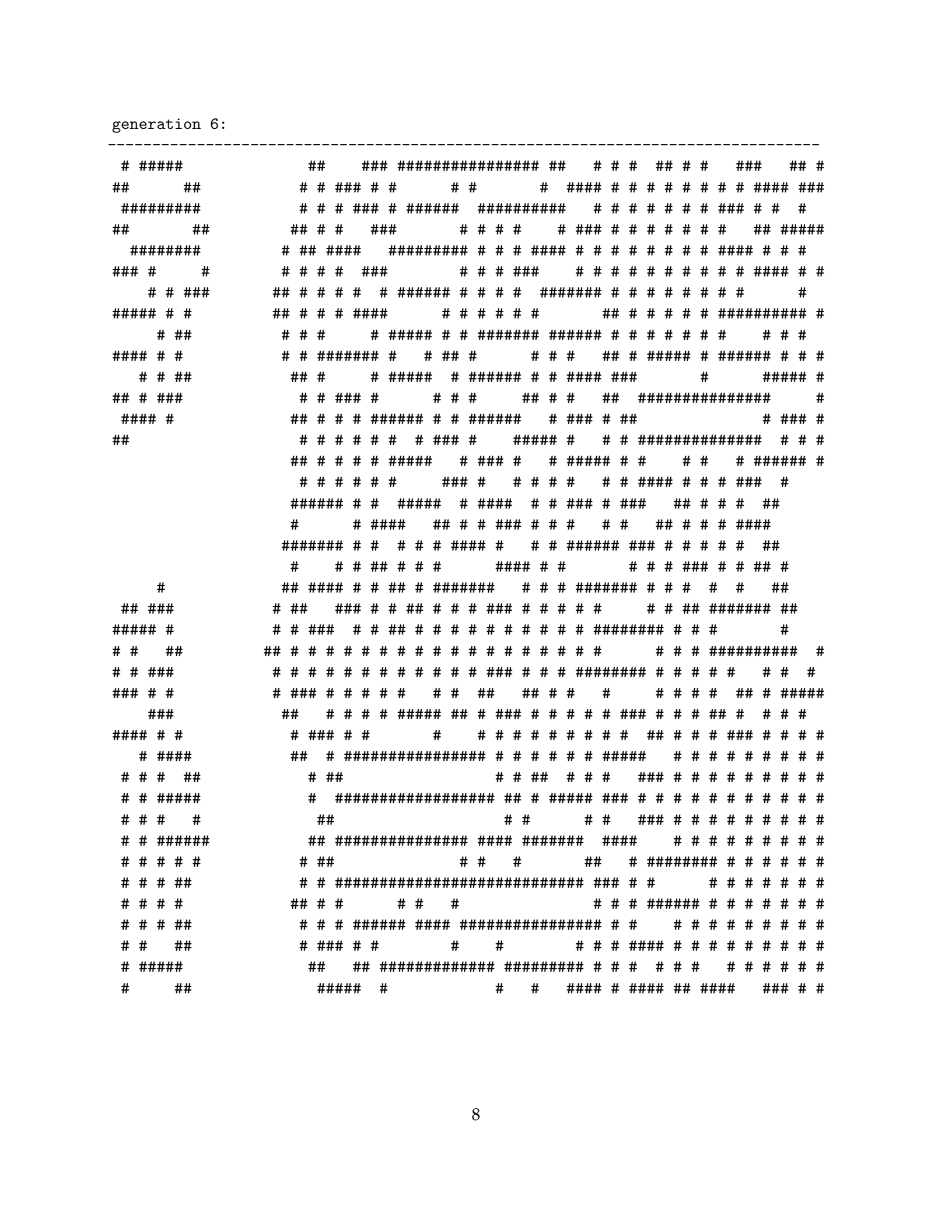
generation 2:

generation 3:

generation 4:

generation 5:

generation 6:



		N						
		16	32	64	128	256	512	1024
T	1	4.353	8.256	21.87	74.43	281.5	1184	5006
	2	4.565	6.818	13.93	42.30	162.4	625.0	2570
	3	2.677	5.322	10.43	33.20	125.7	495.8	2059
	4	2.552	4.221	8.037	25.76	97.80	382.3	1487

Table 2: Wall clock time in μs to calculate a single generation of a $N \times N$ cellular automaton on T threads.

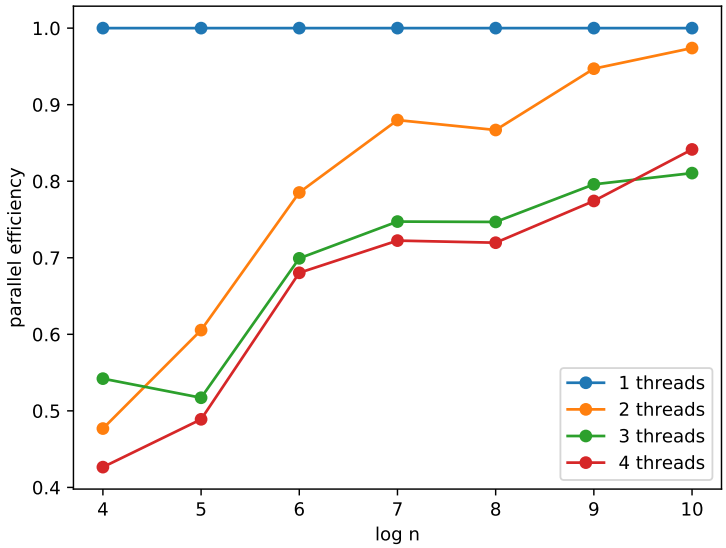


Figure 1: Multithreading efficiency drops as the number of threads increases and the total workload decreases.

Part (b)

See Table 2.

Part (c)

See Figure 1. The work required for thread creation stays constant as the grid grows, while the total work of computing a generational update rises with grid size. Since the former requires more time for more threads, but the latter takes less time with more threads, it should be expected that threading becomes advantageous as the grid size becomes larger.

Problem 3 – A Mersenne unprime

Part (a)

Your code should find there to be 17,984 prime numbers that are less than 2×10^5.

Part (c)

There are—unsurprisingly—no prime numbers less than 10^6 that are factors of M.

Part (d)

There are three prime numbers less than 10^6 that are factors of N: 3, 5, and 41201.

Problem 4 – A multithreaded Sudoku solver

1			9		7			3
	8						7	
		9				6		
		7	2		9	4		
4	1						9	5
		8	5		4	3		
		3				7		
	5						4	
2			8		6			9

(a)

1	2	3						
4	5	6						
7	8	9						
			1	2	3			
			4	5	6			
			7	8	9			
						1	2	3
						4	5	6
						7	8	9

(b)

1								2
	5						4	
		9				6		
			1		2			
				8				
			6		9			
		2				1		
	4						5	
6								9

(c)

Figure 2: Three partial Sudoku grids.

Part (a)

Fig. 3 shows the unique solution to the Sudoku shown in Fig 2a. The code took an average 0.7 ms to compute this solution.

1	6	4	9	5	7	2	8	3
3	8	5	6	2	1	9	7	4
7	2	9	4	3	8	6	5	1
5	3	7	2	8	9	4	1	6
4	1	2	7	6	3	8	9	5
6	9	8	5	1	4	3	2	7
8	4	3	1	9	5	7	6	2
9	5	6	3	7	2	1	4	8
2	7	1	8	4	6	5	3	9

Figure 3: Solution to Sudoku shown in Fig. 2a.

Part (b)

The code computes 283,576 solutions to the Sudoku shown in Fig. 2b.

Part (c)

To extend the recursive algorithm to use OpenMP, we first generate a series of partially filled-in grids by storing the results from a breadth-first search. These grids can then be passed to the recursive algorithm in parallel. The number of grids used to bootstrap the parallelization has an impact on performance; in our experiments we found that 50,000–100,000 grids gave optimal parallel performance for Sudoku 2c, whereas 50–100 grids gave optimal parallel performance for Sudokus 2a and 2b.

Table 3 shows the average wall clock times to compute all solutions to the Sudokus given in Fig. 2.

	Sudoku 2a		Sudoku 2b		Sudoku 2c	
Threads	Time	Efficiency	Time	Efficiency	Time	Efficiency
1	0.81 ms	1.00	0.46 s	1.00	44.6 s	1.00
2	0.49 ms	0.83	0.23 s	0.97	27.3 s	0.89
4	0.27 ms	0.74	0.12 s	0.94	17.2 s	0.71

Table 3: Average wall clock times to compute all solutions to the given Sudokus.

Part (d)

The code computes 4,347,232 solutions to the Sudoku shown in Fig. 2c.

Problem 5 – Testing thread performance on the diffusion equation

Identifying α, β, γ We can write the derivatives $\partial/\partial X$ and $\partial/\partial Y$ in the form

$$\frac{\partial}{\partial X} = \frac{\partial x}{\partial X}\frac{\partial}{\partial x} + \frac{\partial y}{\partial X}\frac{\partial}{\partial y},$$
$$\frac{\partial}{\partial Y} = \frac{\partial x}{\partial Y}\frac{\partial}{\partial x} + \frac{\partial y}{\partial Y}\frac{\partial}{\partial y},$$

which, using the transformation $\boldsymbol{x} = \underline{\underline{T}} \cdot \boldsymbol{X}$, is just

$$\frac{\partial}{\partial X} = T_{11}\frac{\partial}{\partial x} + T_{21}\frac{\partial}{\partial y},$$
$$\frac{\partial}{\partial Y} = T_{12}\frac{\partial}{\partial x} + T_{22}\frac{\partial}{\partial y}.$$

This is equivalent to writing $\boldsymbol{\nabla}_X = \underline{\underline{T}}^{\mathsf{T}} \cdot \boldsymbol{\nabla}_x$, where $\boldsymbol{\nabla}_x$ and $\boldsymbol{\nabla}_X$ denote the gradient operator in each coordinate system. Using this relationship, we can write the Laplacian operator in the

untransformed coordinate system in terms of $\boldsymbol{\nabla}_X$:

$$\begin{aligned}\nabla^2 = \boldsymbol{\nabla}_x \cdot \boldsymbol{\nabla}_x &= \left[\underline{\underline{T}}^{-\mathsf{T}} \cdot \boldsymbol{\nabla}_X\right] \cdot \left[\underline{\underline{T}}^{-\mathsf{T}} \cdot \boldsymbol{\nabla}_X\right], \\ &= \boldsymbol{\nabla}_X \cdot \left[\underline{\underline{T}}^{-1} \cdot \underline{\underline{T}}^{-\mathsf{T}}\right] \cdot \boldsymbol{\nabla}_X, \\ &= \underline{\underline{S}} : (\boldsymbol{\nabla}_X \boldsymbol{\nabla}_X),\end{aligned}$$

where $\underline{\underline{S}}$ is the symmetric matrix $S_{ij} = T^{-1}_{ik} T^{-1}_{jk}$. When the components of $\underline{\underline{S}}$ are denoted

$$\underline{\underline{S}} = \begin{pmatrix} \alpha & \beta/2 \\ \beta/2 & \gamma \end{pmatrix},$$

this expression takes the form

$$\boxed{\nabla^2 = \alpha \frac{\partial^2}{\partial X^2} + \beta \frac{\partial^2}{\partial X \partial Y} + \gamma \frac{\partial^2}{\partial Y^2}.}$$

Now, since the inverse transformation is given by

$$\underline{\underline{T}}^{-1} = \frac{1}{D} \begin{pmatrix} T_{22} & -T_{12} \\ -T_{21} & T_{11} \end{pmatrix},$$

where $D = T_{11}T_{22} - T_{12}T_{21}$ is the determinant of $\underline{\underline{T}}$, the coefficients α, β, γ are linked to the transformation matrix by

$$\boxed{\alpha = T^{-1}_{1i} T^{-1}_{1i} = \frac{1}{D^2}\left(T_{22}^2 + T_{12}^2\right),}$$

$$\boxed{\beta = 2T^{-1}_{1i} T^{-1}_{2i} = -\frac{2}{D^2}\left(T_{22}T_{21} + T_{11}T_{12}\right),}$$

$$\boxed{\gamma = T^{-1}_{2i} T^{-1}_{2i} = \frac{1}{D^2}\left(T_{11}^2 + T_{21}^2\right).}$$

Part (a)

Defining $\nu = \Delta t / h^2$ and $A = 4\alpha\nu$, $B = \beta\nu$, $C = 4\gamma\nu$, our discretized update takes the form

$$\begin{aligned}u^{n+1}_{j,k} = \left[1 - \frac{A+C}{2}\right] u^n_{j,k} + \frac{A}{4}\left[u^n_{j-1,k} + u^n_{j+1,k}\right] + \frac{C}{4}\left[u^n_{j,k+1} + u^n_{j,k-1}\right] \\ + \frac{B}{4}\left[u^n_{j+1,k+1} - u^n_{j-1,k+1} - u^n_{j+1,k-1} + u^n_{j-1,k-1}\right].\end{aligned}$$

When we substitue in $u^n_{j,k} = \lambda^n_{l,m} e^{i(jl+km)h}$ and divide through by $\lambda^n_{l,m} e^{i(jl+km)h}$, we obtain an expression for the amplification factor $\lambda_{l,m}$ of the (l, m) Fourier mode,

$$\begin{aligned}
\lambda_{l,m} &= \left[1 - \frac{A+C}{2}\right] + \frac{A}{2}\left[\frac{e^{-ilh} + e^{ilh}}{2}\right] + \frac{C}{2}\left[\frac{e^{-imh} + e^{imh}}{2}\right] \\
&\qquad + \frac{B}{2}\left[\frac{e^{i(l+m)h} + e^{-i(l+m)h}}{2} - \frac{e^{i(m-l)h} + e^{i(l-m)h}}{2}\right], \\
&= \left[1 - \frac{A+C}{2}\right] + \frac{A}{2}\cos(lh) + \frac{C}{2}\cos(mh) + \frac{B}{2}\Big[\cos\left[(l+m)\,h\right] - \cos\left[(l-m)\,h\right]\Big], \\
&= (1-A) + \frac{A}{2}\Big[\cos(lh) + \cos(mh)\Big] + \frac{B}{2}\Big[\cos\left[(l+m)\,h\right] - \cos\left[(l-m)\,h\right]\Big] \\
&\qquad + \frac{C-A}{2}\Big[\cos(mh) - 1\Big].
\end{aligned}$$

Rewriting as q, r Now consider two trigonometric identities. First, since

$$\begin{aligned}
\cos 2\xi + \cos 2\eta &= \cos^2\xi - \sin^2\xi + \cos^2\eta - \sin^2\eta, \\
&= (\cos^2\xi - \sin^2\xi)(\cos^2\eta + \sin^2\eta) \\
&\qquad + (\cos^2\eta - \sin^2\eta)(\cos^2\xi + \sin^2\xi), \\
&= 2(\cos^2\xi\cos^2\eta - \sin^2\xi\sin^2\eta), \\
&= 2(\cos\xi\cos\eta + \sin\xi\sin\eta)(\cos\xi\cos\eta - \sin\xi\sin\eta), \\
&= 2\cos(\xi-\eta)\cos(\xi+\eta),
\end{aligned}$$

we can rewrite

$$\cos(lh) + \cos(mh) = 2\cos\left[\frac{(l+m)h}{2}\right]\cos\left[\frac{(l-m)h}{2}\right].$$

Secondly, because

$$\begin{aligned}
\cos 2\zeta &= \cos^2\zeta - \sin^2\zeta, \\
&= \cos^2\zeta - \left(1 - \cos^2\zeta\right), \\
&= 2\cos^2\zeta - 1,
\end{aligned}$$

we can rewrite

$$\cos[(l+m)h] - \cos[(l-m)h] = 2\left\{\cos^2\left[\frac{(l+m)h}{2}\right] - \cos^2\left[\frac{(l-m)h}{2}\right]\right\}.$$

Applying these to our expression above, assuming $\alpha = \gamma \implies A = C$, and introducing $q = \cos[(l+m)h/2], r = \cos[(l-m)h/2], \hat{B} = B/A$, the amplification factor becomes

$$\lambda_{l,m} = 1 + A\left[qr - 1 + \hat{B}(q^2 - r^2)\right].$$

Extrema on [-1,1]^2 The set of all Fourier modes $(l, m) \in \mathbb{Z}^2$ maps onto the square $(q, r) \in [-1, 1]^2$, so we conclude that the method is unconditionally unstable if

$$\max_{q,r\in[-1,1]^2} f_{\hat{B}}(q, r) > 0,$$

where we havve introduced the function $f_{\hat{B}}(q, r) = qr - 1 + \hat{B}(q^2 - r^2)$. By observation, this relationship can only be satisfied by nonzero $\hat{B}$. The function's Hessian matrix has a determinant which is always negative:

$$\underline{\underline{H}} = \boldsymbol{\nabla}_q \boldsymbol{\nabla}_q f = \begin{pmatrix} 2\hat{B} & 1 \\ 1 & -2\hat{B} \end{pmatrix} \implies \det\left(\underline{\underline{H}}\right) = -(1 + 4\hat{B}^2),$$

indicating it has no local extrema, and its maxima and minima must lie on the boundaries of the square $(q, r) \in [-1, 1]^2$. Since the function has the symmetries $f_{\hat{B}}(q, r) = f_{\hat{B}}(-q, -r)$, $f_{\hat{B}}(q, r) + 1 = -(f_{-\hat{B}}(-q, r) + 1)$, and $f_{\hat{B}}(q, r) = f_{-\hat{B}}(r, q)$, we can perform our analysis without loss of generality by choosing $q = 1$ and $\hat{B} > 0$. Doing so, we obtain a function of one variable

$$g(r) = f(1, r) = r - 1 + \hat{B}(1 - r^2),$$

which has a local extremum at $r_* = 1/2\hat{B}$ with value

$$g(r_*) = \frac{1}{4\hat{B}} + \hat{B} - 1.$$

There are two regions of interest for positive $\hat{B}$:

- **Case 1:** $0 < \hat{B} \leq 1/2$. Since $r_* > 1$, the local extremum lies outside of our domain of interest. The maximum of g on the $q = 1$ boundary is determined by the function values at the endpoints, $g(1) = 0$ and $g(-1) = -2$. The criteria for unconditional instability is not satisfied.
- **Case 2:** $\hat{B} > 1/2$. At $\hat{B} = 1/2$, the maximum value occurs at the corner $r_* = 1$ and takes the value $g(r_*) = 0$. In this region,

  $$\frac{\partial}{\partial \hat{B}}\left[g(r_*)\right] = 1 - \frac{1}{4B^2} > 0 \text{ for all } \hat{B} > \frac{1}{2},$$

  and $0 < r_* < 1$ for all $\hat{B} > 1/2$.

This implies that for all $\hat{B} > 1/2$, the maxmimum value $g(r_*) = f(1, r_*) > 0$ and is located within the region $(q, r) \in [-1, 1]^2$. The criteria for unconditional instability is satisified.

This analysis can be extended to negative $\hat{B}$ by symmetry arguments—consider, for instance, the analogous consideration of the $r = 1$ boundary with $\hat{B} < 0$. After considering all four boundaries, we conclude that for $|\hat{B}| > 1/2$, the method is unconditionally unstable. This corresponds to the following two statements:

$$\boxed{|\beta| > 2\alpha} \implies \text{unconditionally unstable},$$

$$\beta \leq 2\alpha \implies -2 \leq f(q, r) \leq 0 \text{ for all } (q, r) \in [-1, 1]^2.$$

Geometric interpretation Consider what this means in terms of the inverse transformation matrix $\underline{\underline{T}}^{-1}$. Each row of this matrix is a unit vector of the transformed coordinate system: $\underline{\underline{T}}^{-1} = \left(\hat{\boldsymbol{X}}, \hat{\boldsymbol{Y}}\right)^{\mathsf{T}}$. With this notation, the constraints we derived above are just

$$\alpha = \gamma \implies |\hat{\boldsymbol{X}}| = |\hat{\boldsymbol{Y}}|$$

$$\left|\frac{\beta}{2}\right| \leq \alpha = \gamma \implies |\hat{\boldsymbol{X}} \cdot \hat{\boldsymbol{Y}}| \leq |\hat{\boldsymbol{X}}|^2 = |\hat{\boldsymbol{Y}}|^2.$$

We see that our β-condition is simply a rewriting of the Cauchy–Schwarz Inequality! In other words, the method is unconditionally unstable if used with a β that does not correspond to a coordinate transformation $\underline{\underline{T}}$.

A side note about a slight abuse of notation. We refer to $\hat{\boldsymbol{X}}$ and $\hat{\boldsymbol{Y}}$ as unit vectors, since they correspond to $(1, 0)$ and $(0, 1)$ in the transformed coordinate system. Yet in the above, we treat them as though their magnitudes are not 1. This is because the inner product and norms in the above calculation are those of the untransformed coordinate system, NOT the transformed coordinate system. Thus, $|\hat{\boldsymbol{X}}|$ is not necessarily 1. Instead, as written before, $|\hat{\boldsymbol{X}}| = (T_{1i}^{-1} T_{1i}^{-1})^{1/2}$.

We can explicitly denote the inner product of the transformed coordinate system like $\langle \cdot, \cdot \rangle_X$. For instance, $\left\langle \hat{\boldsymbol{X}}, \hat{\boldsymbol{X}} \right\rangle_X = \left\langle \hat{\boldsymbol{Y}}, \hat{\boldsymbol{Y}} \right\rangle_X = 1$. The fact that unit vectors in one system are not necessarily of magnitude 1 in the other coordinate system indicates the coordinate transform involves a rescaling of lengths.

Conditional Stability Recall that the amplification factor is given by

$$\lambda_{l,m} = 1 + A f(q, r),$$

and that if $|\beta| \leq 2\alpha$, we have $-2 \leq f(q, r) \leq 0$ for all $q, r \in [-1, 1]^2$. Accordingly, the method is conditionally stable ($|\lambda_{l,m}| \leq 1$) if $A < 1$. This corresponds to

$$\text{stable if } \nu < \frac{1}{4\alpha} \implies \boxed{\Delta t < \frac{h^2}{4\alpha}.}$$

Part (b)

Below are the wall clock time and efficiency recorded after integrating the diffusion equation in the transformed coordinate system for 1000 timesteps, using a variety of grid sizes N and numbers of threads P. For small N, efficiency falls quickly as threads are added, since gains from parallelization are offset by the overhead of launching more threads. At larger N, since the gains from parallelization are proportionally larger while the overhead stays the same, efficiency falls more slowly as threads are added.

Part (c)

As shown in Fig. 5, the standard deviation in both systems decreases exponentially with time as the system diffuses away to some uniform state. In the transformed system, this diffusion happens on a slightly faster time scale than in the untransformed case.

Recall that diffusion of strength D with regard to a length scale L happens on a time scale $\tau \sim L^2/D$—this comes directly from $\partial_t = D\partial_x^2$. Now, consider that fact that (as explained in the

		P							
		1	2	3	4	5	6	7	8
N	64	0.0289 s	0.0198 s	0.0157 s	0.0132 s	0.0103 s	0.0100 s	0.0095 s	0.0093 s
	128	0.081 s	0.056 s	0.041 s	0.033 s	0.027 s	0.024 s	0.028 s	0.025 s
	256	0.236 s	0.175 s	0.125 s	0.144	0.079 s	0.070 s	0.061 s	0.057 s
	512	0.94 s	0.63 s	0.47 s	0.42 s	0.31 s	0.25 s	0.23 s	0.20 s
	1024	3.61 s	1.99 s	1.63 s	1.32 s	1.08 s	0.82 s	0.81 s	0.77 s
	2048	14.7 s	7.8 s	6.4 s	4.6 s	4.0 s	3.2 s	3.1 s	2.6 s
	4096	59.8 s	34.2 s	27.3 s	19.0 s	16.5 s	12.9 s	12.7 s	10.7 s

Table 4: Wall clock time to simulate 1000 time steps.

		P							
		1	2	3	4	5	6	7	8
N	64	1	0.73	0.61	0.55	0.56	0.48	0.43	0.39
	128	1	0.72	0.66	0.62	0.60	0.55	0.41	0.40
	256	1	0.67	0.63	0.41	0.60	0.57	0.56	0.52
	512	1	0.74	0.67	0.56	0.61	0.62	0.60	0.58
	1024	1	0.91	0.74	0.68	0.67	0.73	0.64	0.59
	2048	1	0.94	0.76	0.80	0.73	0.77	0.69	0.72
	4096	1	0.87	0.73	0.79	0.73	0.77	0.67	0.70

Table 5: Threading efficiency after simulating 1000 time steps.

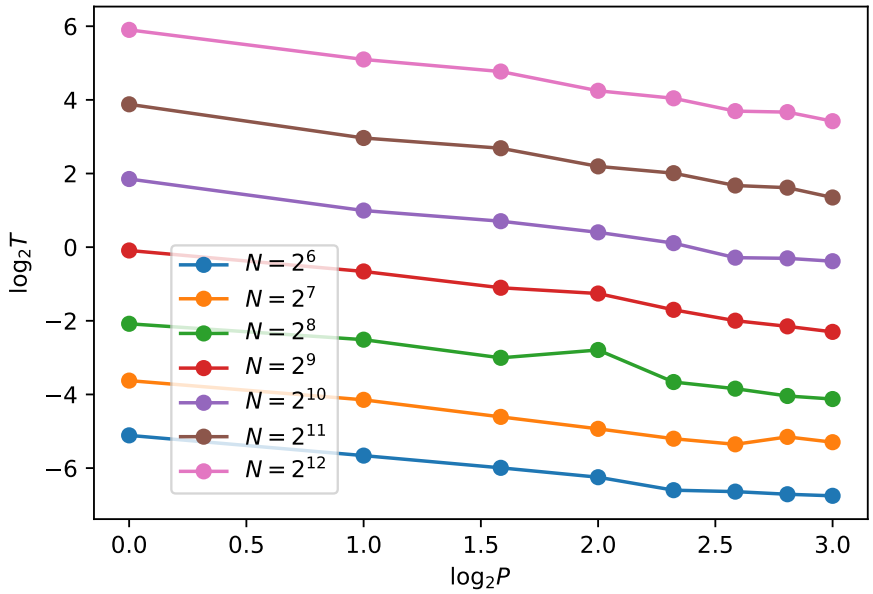


Figure 4: A plot of the number of threads against wall clock time required to simulate 1000 time steps.

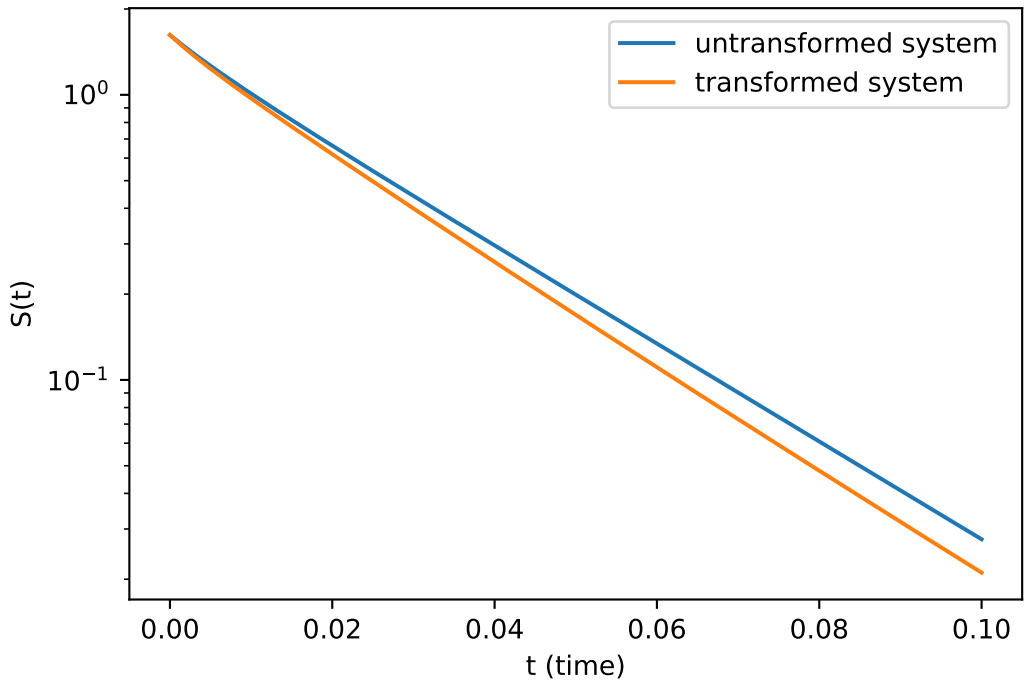


Figure 5: The standard deviation $S(t)$ decreases more quickly in the transformed system.

geometric interpretation section), the coordinate transform introduces a rescaling of length. This alters the scaling of our derivatives such that

$$\frac{\partial}{\partial x} \sim \frac{\partial}{\partial y} \sim \frac{1}{L} \implies \frac{\partial}{\partial X} \sim \frac{1}{L_X} = \frac{|\hat{\boldsymbol{X}}|}{L}, \frac{\partial}{\partial Y} \sim \frac{1}{L_Y} = \frac{|\hat{\boldsymbol{Y}}|}{L}.$$

This corresponds to an effective rescaling of the time scale, depending on the orientations of inhomogeneities in the diffusing field. For instance, given the example here, fields in the transformed system varying only in Y will diffuse at the same speed as in the untransformed system, since $|\hat{\boldsymbol{Y}}| = 1 \implies L = L_Y \implies$ no rescaling of lengths. But fields varying in the X direction will diffuse more quickly, as $|\hat{\boldsymbol{X}}| > 1 \implies L_X = L/|\hat{\boldsymbol{X}}| < L$. Lengths in this direction—and therefore the timescale of diffusion of inhomogeneities in this direction—have been effectively reduced.

Part (d)

We can extend to arbitrary $\alpha \neq \gamma$ by applying our geometric interpretation. The system will be unconditionally unstable unless it corresponds to a possible coordinate transform matrix, any of which will satisfy the Cauchy–Schwarz inequality. Therefore, we require

$$\langle \hat{\boldsymbol{X}}, \hat{\boldsymbol{Y}} \rangle \leq |\hat{\boldsymbol{X}}||\hat{\boldsymbol{Y}}| \implies \frac{\beta}{2} \leq \sqrt{\alpha\gamma}.$$

Analogously,

$$\Delta t < \frac{h^2}{2}\left(\frac{1}{\alpha+\gamma}\right),$$

stipulating that we must limit our time step to capture information propogating at the speed of diffusion in each of the transformed coordinate directions.