Topology Project: Research Notes

Alexandre Champagne-Ruel

May 20, 2025

Contents

1 Previous notes from Cole

{sec:previous-notes}

1.1 May 12th, 2022

I'm starting a notebook because I keep forgetting what I'm working on. Hopefully this will complement the git logs.

Today I was able to update the `Ensemble` constructor function and write a function in `explore-parameters.jl` that will generate an Ensemble and run the dynamics. I need to check for bugs and work on the visualization side now.

I need to make sure to use `@quickactivate` with DrWatson; otherwise, saving does not work as expected.

1.2 June 24th, 2022

I should focus on this project a bit more. My goal today is to make a plan of attack. This should include a few different things, mostly issues on git to guide future development.

Main goals:

- Parameter sweep of the well-mixed case:
  - Plot Max MA, MA distribution fit, integer number against (forward, backward rate)
- Parameter sweep of spatially distributed cases:
  - Plot same things against diffusion rates
  - Start with Line Graphs, then lattice, then random.
  - Compute MA of Graphs. (Pick forward/backward rate from sweep above and vary diffusion)

To do those we'll need a coherent data management system and an efficient analysis pipeline.

1.3 July 6th, 2022

Taking a look after a couple of days away. The Simulation class is implemented now. I think the main thing to do is convert the .bson files to a .csv (somewhat efficiently) and then write some R scripts to do the analysis we want.

The save system is working; now it's time to write the analysis pipeline.

1.4 July 12th, 2022

Analysis pipeline mostly sorted out. There's likely some bugs lurking about, and I don't know what will happen at scale, but we'll see. The big problem now is mostly a problem for the future: once specific compounds are stabilized by different reactors, it will be difficult to keep track of all the information in coherent ways. The same problem exists for simulations with different rate constants in different reactors.

Need to check the assumptions of the simulation.

1.5 July 13th, 2022

Lots of systems in place now. Just debugging the simulations before running the parameter sweep. Something disturbing is the degree to which the reactors without inflow seem to be completely static for all time. This strikes me as a serious bug. I can't seem to find the source.

1.6 September 9th, 2024 (Alex)

- Seems like Cole has fixed the function call in `test_timing.jl`. I have launched several simulations varying the parameter set, and it executes normally.
- I am now trying to get the parameter explorations to work (`test_timing.jl` and `explore-test.jl`), ideally using multiple CPU cores. The original file coded by Cole (`explore-test.jl`) was still looping sequentially over the different parameters, which is why I have made a new version (`explore-test.jl`) that computes the different parameter permutations in advance and launches the simulations using these pre-computed parameters. It seems to use the full extent of the multiprocessing module, as observed from CPU usage in e.g., `htop`.

1.7 September 11th, 2024 (Alex)

- Did a bunch of tests in the past few days—`test_timing.jl` and `explore-test.jl` are now working on Sol, the GCloud cluster, etc.

1.8 September 12th, 2024 (Alex)

- Started coding a minimal working example: plotting the population time series.

2 Description of the system

2.1 Artificial chemistry

{sec:desc-system}

{subsec:artificial-chemistry}

Basically, we want to use a system based on artificial chemistry to investigate the conditions promoting the formation of complex objects—where, by "complex objects", we refer to molecular complexity. Specifically, we will be focusing our analysis on investigating the spatial topology of the environment, to try to determine how it can program the emergence of complexity. We will utilize Assembly Theory to quantify the complexity formed therein, and in doing so this model will constitute a new test environment. Among other things, this could help refine our methods of calculation for the Assembly Index, and inform our knowledge on how the Assembly Index responds to various parameters and/or is correlated to life-like properties.

Artificial chemistry implementations can be categorized along two axis: how abstract (or realistic) they represent "real" chemistry, and whether they aim to model early or late evolution (see Fig. 1).

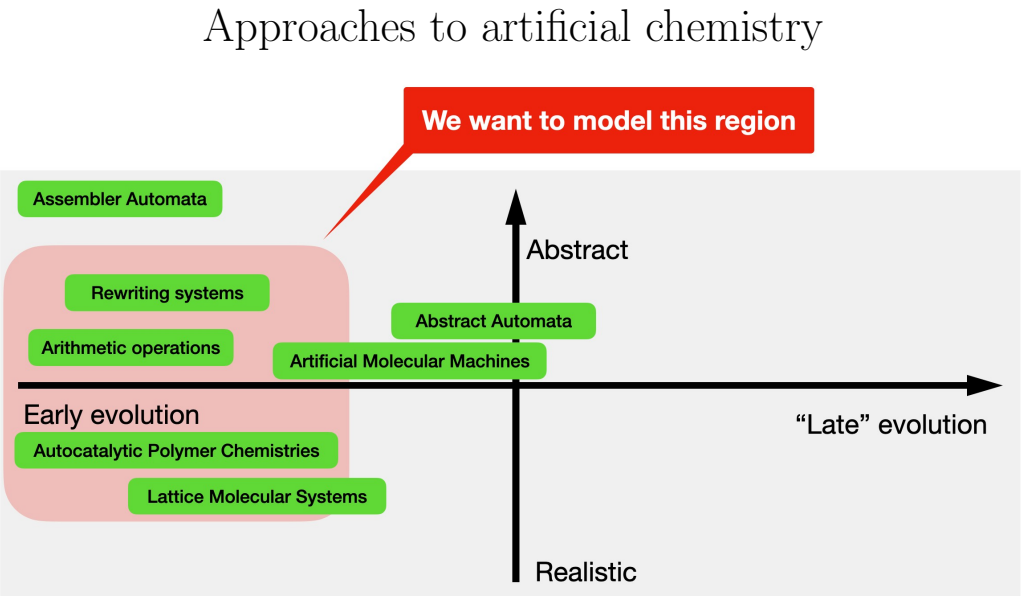

Figure 1: Classification of AC approaches along two axis: abstract vs realistic, and early vs late evolution. In the context of this project we will be focusing on the early evolution, using a balanced approach between more abstract and more realistic models.

{fig:abstraction-stage}

The underlying (artificial) chemistry is based on several transformations: formard reactions, backward reactions, diffusion (Table 1). These reactions are parametrized by rate coefficients (k_f, k_b, k_d). Several other parameters can be defined, such as simulations (temporal) length τ_{max}, the number of reactors N, the number of inflows n, total mass M, etc.

Backward Reaction Rate. We have chosen to keep the backward reaction rate *constant* with respect to molecule length (i.e., integer value). Both alternatives can be justified: larger molecules might break more easily due to having more bonds, or they might be harder to break because they are more stable. Here, we adopt the former assumption.

Forward reaction	$A + B \xrightarrow{k_f} C$
Backward reaction	$C \xrightarrow{k_b} A' + B$
Diffusion	$C \xrightarrow{k_d} \varnothing$
	$C_i \xleftrightarrow{k_d} C_j$

Table 1: Reactions occurring in the artificial chemistry implemented in this system. Forward (constructive) reactions happens at rate k_f, backward (destructive) reactions happen at rate k_b and diffusion (where a molecule is shifted either to the next chemostat or out of the system) happens at rate k_d. Mass M is fixed, so the in-flow is coupled with the diffusion parameter.

{tab:reactions}

2.2 System of reactors

{subsec:system-of-reactors}

These transformations are applied on a population of integers. These integers form a well-mixed system, inside a reactor/chemostat (Fig. 2a). The number of integers #1 is fixed, which translates into a fixed in-flow in the first reactor (as these integers are used to form other, more complex integers, or diffuse to the next chemostats). The system as a whole consists of several of these reactors, coupled together via in- and out-flows (defined by the corresponding rates), in a specific topology. For example, one of the simplest topology is that of the regular lattice (Fig. 2b). When diffusion is very high, the whole system becomes well-mixed.

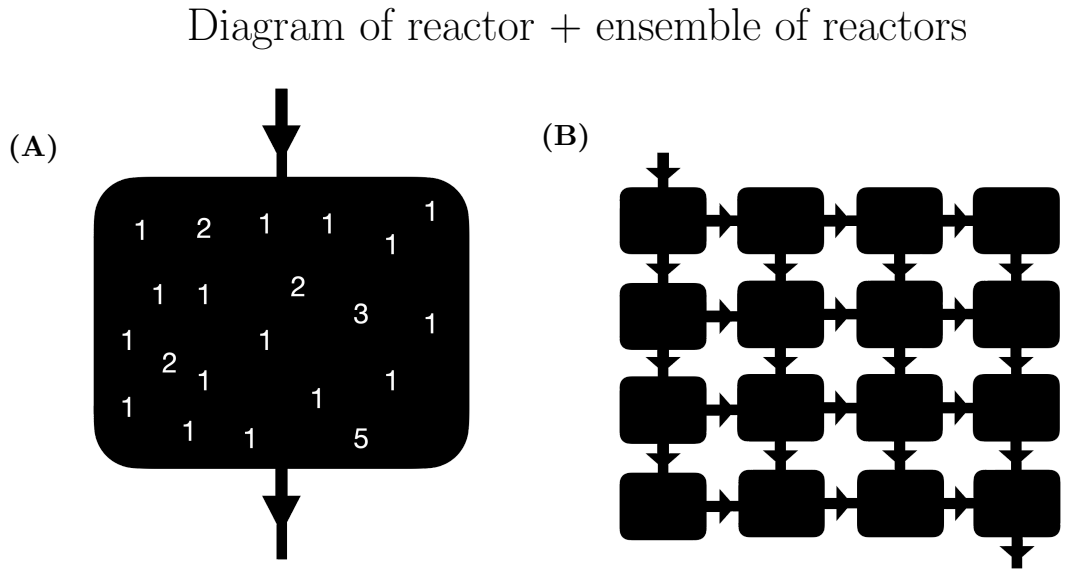

Figure 2: **(A)** Single reactor, inside of which a bunch of integers react. In-flow is fixed, i.e. the # of ones is fixed, and these react according to the reactions described in Table 1. An out-flow connects the reactor to subsequent chemostats. **(B)** Network of reactors/chemostats. Displayed is a regular lattice (note that diffusion happens both ways). There is a single in-flow, and a single out-flow. Other topologies are possible (see Fig. 3 below.)

{fig:reactor-ensemble}

2.3 Topologies

{subsec:topologies}

Several topologies can be used to connect the reactors together (Fig. 3). Examples include: path, lattice, Erdos-Renyi (random), regular (uniform node degree) or Barabasi-Albert (power-law degree distribution). This is especially relevant given that we're studying living systems, whose networks have been shown to possess certain specific properties deriving from their topology (e.g., resilience from BA networks, etc.) Other topologies (e.g. lattice, random) can be used as control/neutral.

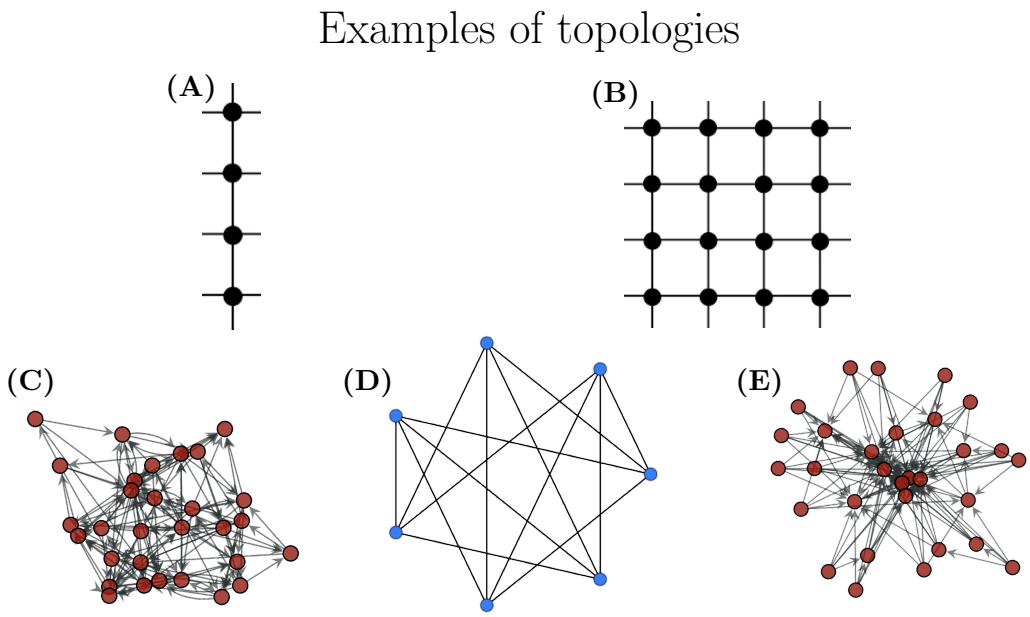

Figure 3: Illustrations of the different topologies we'll be exploring (at least, in the near future). The chemostats can be connected according to any of these topologies, which we presume will affect the construction of molecular complexity in the mixture. **(A)** Simple path lattice. **(B)** Lattice. **(C)** Random (Erdos-Rényi) graph. **(D)** Regular graph (all nodes have the same degree). **(E)** Scale-free (Barabasi-Albert) graph, with the distribution of nodes follow a power law.

{fig:topologies}

2.4 Integer chemistry and Assembly Theory

{subsec:integer-chemistry}

We'll be using Assembly Theory to quantify the complexity emerging from the mixture, therefore we'll use the Assembly Index (A). Cole has calculated A for integers < 10000. The Assembly Index increases as the logarithm of integer values (Fig. 4). We'll however probably be using both, as some relationships are better illustrated using y-axis as integers, and others with the y-axis as the Assembly Index.

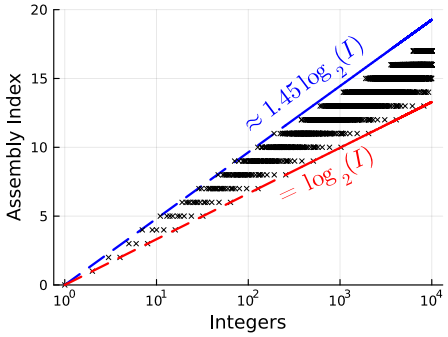

Figure 4: Assembly Index A for integers < 10000. A increases as the logarithm of integer values.

{fig:integers-assembly}

3 Experiments with the stochastic method

3.1 MS01: time series

{sec:experiments}

{subsec:MS01}

A first result we can plot using this system is the time series of the populations (shown on Fig. 5). The layout of this figure follows the same structure as the one on Fig. 2b: each panel shows the evolution of integers $I \in 1...10$ through time. The blue curve shows ones, we can see the number of ones is fixed on the first panel (top left). Diffusion pushes molecules to the next chemostats that are connected to the first one (where the in-flow is) so we're seeing initial transients/peaks at different times. The higher we increase diffusion, the faster these peaks happen, and the shorter is the initial transient.

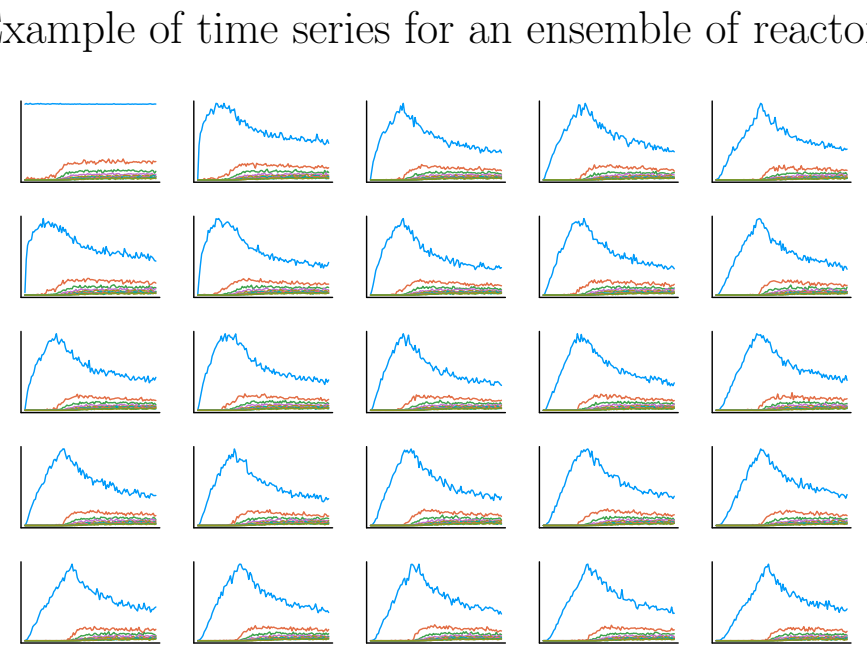

Figure 5: Time series for a simulation of 25 chemostats connected in a regular lattice. The first panel (top left) corresponds to the chemostat with an in-flow connected to it. The ten first species (integers $I \in 1...10$) are shown.

{fig:time-series}

3.2 MS02: varying the diffusion rate

A first parameter we're exploring is the diffusion rate. We've sampled simulations with diffusion rates $k_d \in [10^{-6}, 10^2$ and plot the first ten species (excluding ones) on Fig. 9. Panel A shows the average populations across reactors, i.e. taking all the chemostats together as one single population and calculating the average for species S at the last iteration of the simulation τ_{max}. We've done this calculation for 100 values of k_d (without using statistical ensembles for now). The shaded area on Panel A represents the standard deviation across chemostats, and is also illustrated on Panel B. We've also added a vertical (dashed) line to indicade the forward rate k_f. {subsec:MS02}

One thing we can readily notice from Fig. 6 is the presence of transitions. On Panel A we have four regimes: low values in the populations of complex (i.e. $I > 1$) species for lower values of the diffusion coefficient ($k_d < k_f$), then increasing complexity ($k_f < k_d < 1$), then a dip in complexity (k_d 10), and finally complexity increasing again for very high diffusion rates ($k_d \gtrsim 10^2$). Panel B shows an increase in σ until k_d 10^{-1} then a sharp decrease ($10^{-1} < k_d < 10$).

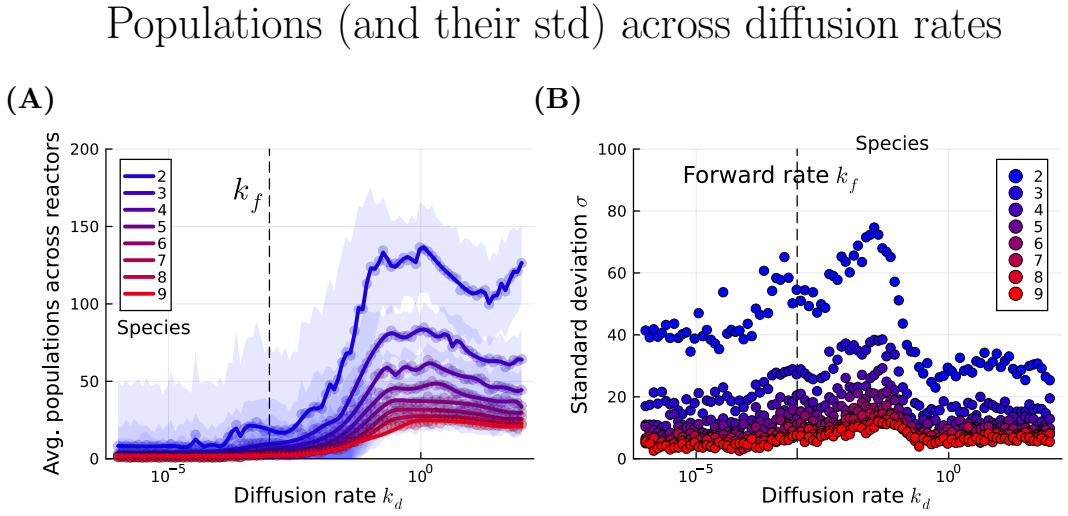

Figure 6: **(A)** Average populations, calculated across all chemostats, at τ_{max} for the first ten integers excluding ones. Shaded area represents the standard deviation (also pictured on Panel B). **(B)** Standard deviation for the same species shown on Panel A. {fig:prelim-diffusion}

Another thing we can calculate from these systems is the total mass, i.e. $I_i \times f_i$ (where I_i is the integer of species i and f_i the number of identical copies or "copy number"). Shown on Fig. 6 is the total mass M against the same variation in coefficient discussed above. M stays constant for most values of the diffusion coefficient but increases exponentially for the highest values of k_d.

Total mass of the system vs diffusion coefficient

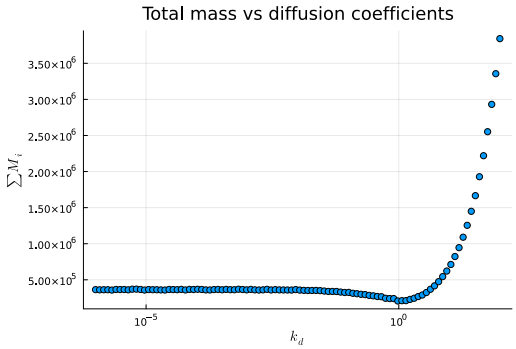

Figure 7: Increase in total mass as we increase diffusion. Total mass M is calculated by multiplying integers I_i with their copy number f_i.

{fig:prelim-mass}

Putting this all together: There seems to be three regimes based on what we're seeng in Figs 6-7: initially, we have a well-mixed system in the first chemostat. The complexity in chemostat #1 increases, but all other chemostats remain empty, which is why we're seeing low average values in Fig. 6 (they're calculated over all chemostats). Then, as we increase diffusion the other chemostats are progressively filled up: this is why the standard deviation increases. Finally, as all chemostats are fully filled and we continue increasing the diffusion, the system returns to a well-mixed state and the mass increases rapidly (because complexity now builds up in all the chemostats).

3.3 MS03: measuring distance from source

One additional experiment we've done was to investigate how the mean assembly index (per reactor) varied with the distance from source D. For example, the distance D between the source (in green) and the target chemostat (in red) shown on Fig. 8a would be six. We've evaluated how the complexity varied with D for three diffusion regimes: low diffusion ($k_d = 10^{-4}$), intermediate diffusion ($k_d = 10^{-2}$) and high diffusion ($k_d = 10$). Complex molecules remain in the chemostats that are located close to the source $D = 1, 2$ when diffusion is low (Fig. 8, blue curve) or intermediary (red curve). When diffusion is increased (green curve), complexity spreads to the other reactors. {subsec:MS03}

However, there remains a degeneracy when we examine the complexity contained in chemostats located at identical distances from the source (Fig. 8, dashed box surrounding data points at $D = 5$). Multiple chemostats at $D = 5$ have different (average values of the) assembly index. This is clearly evident when looking at the inset in Fig. 8 that displays the density plots (i.e., reconstructed PDF): the heavy tails extending to high values of A clearly differ from one chemostat to another. Consequently, D alone does not fully determine the quantitative complexity of the mixture.

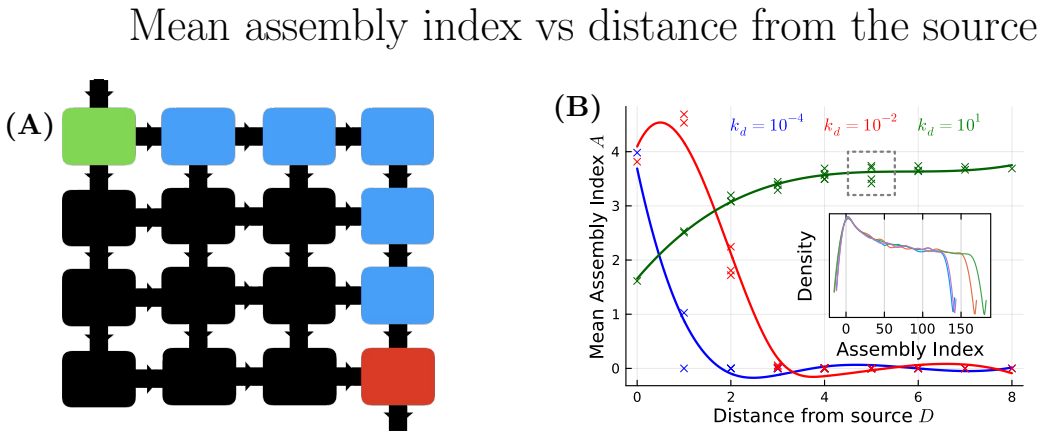

Figure 8: **(A)** Here we investigate how complexity varies as a function of the distance D from the source. For example, the distance between the in-flow (green) and the target chemostat (red) is six. **(B)** Complexity as a function of the distance from the source, for three diffusion regimes. Inset: density plots for the four chemostats having $D = 5$. Clearly, D alone does not fully determine the complexity of the mixture. {fig:prelim-distance}

3.4 MS04: measuring detection thresholds

{subsec:MS04}

Next, we want to relate these experiments to observational missions by evaluating thresholds in complexity that we could detect. We can therefore evaluate the abundance of the different species (i.e., populations as a fraction of total # of molecules or total mass—here we chose the former) and determine the most complex molecule we could detect using an instrument capable of measuring concentrations of 10^{-5} (i.e. 1 molecule in a system having a total of 10^5 molecules), then 10^{-4}, and so on. Fig. 9 how the complexity of detectable molecules varies as we change the diffusion coefficient.

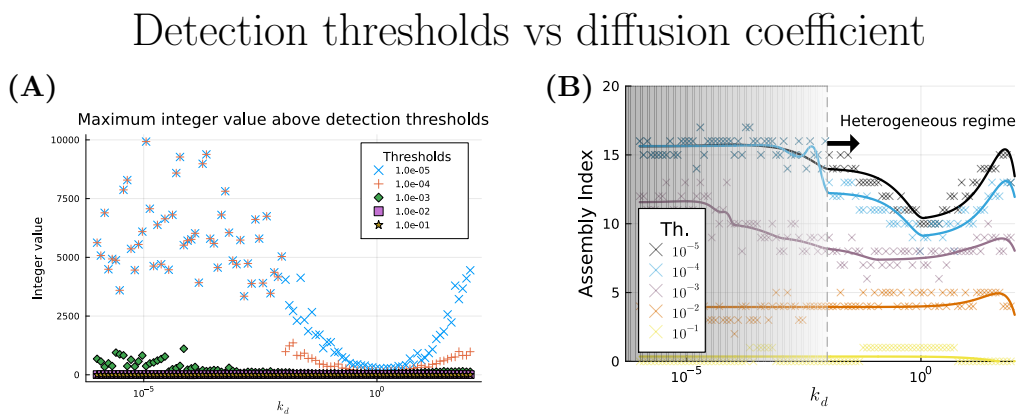

Figure 9: **(A)** Most complex detectable molecule for different thresholds (indicated in the legend) as we vary the diffusion rate. Raw integer values are shown. **(B)** Same as Panel A, but the Assembly Index is used instead to indicate the complexity of the molecules.

{fig:prelim-abundance}

In Fig. 9, on Panel A we plot the raw integer values against k_d whereas on Panel B we're showing the assembly index instead. We're using five different detection thresholds $(10^{-5}, \cdots, 10^{-1})$. Each data point indicates the integer value (or assembly index) of the most complex molecule that's detectable using this specific threshold. Going back to what we mentioned previously about transitions and regimes, we can now clearly see the three regimes we talked about in Fig. 9: for lower values of k_d, all the complexity is contained in the first chemostat and we're just sampling the heavy tails, as we increase diffusion the molecules spread in the other chemostats and complexity suddenly decreases, finally as we further increase k_d complexity rises again.

Basically, we have two competing phenomena: (1) increasing complexity as we increase k_d, but also (2) decreased ability to detect the molecules when they spread initially (which decreases their concentration)—until they have filled all the chemostats and their concentration increases again. This is why we're seeing a minima on Fig. 9: this point represents the area of the parameter space where the molecules are in the process of filling up the chemostats.

Upcoming experiments Some things we are considering:

- making a figure of the "cones" which define assembly spaces for graphs/integers/-molecules (similar to the one in Fig. 10 below)
- decoupling in-flow and diffusion: right now in-flow is implicit, i.e. we have a fixed number of ones in chemostat #1, and when we increase diffusion we push molecules

to the next chemostats (which amounts to refilling chemostat #1, since the number of ones is fixed). This prevents us from increasing the in-flow independently from the diffusion coefficient, which could increase complexity without increasing diffusion so much (making the whole system a giant well-mixed system, and also making the simulations much longer to compute)

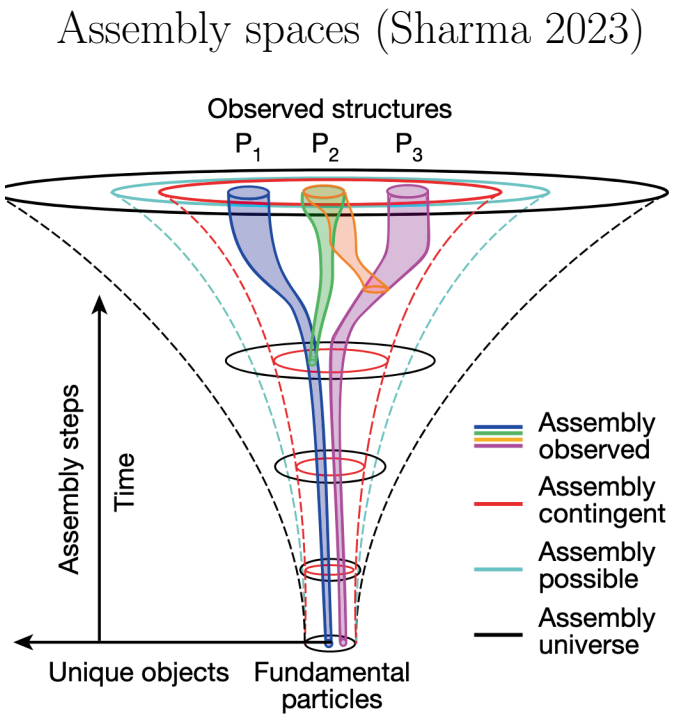

Figure 10: Cones defining the different assembly spaces. Taken from Sharma et al. (2023). {fig:assembly-cones}

4 Experiments with the tau-leaping method (bug with inflow)

4.1 MS06: implementing tau-leaping

From the draft of the paper: {subsec:MS06}

Let k_i denote one of the reaction rates. The propensity a_i for a specific reaction to occur is determined by multiplying the reaction rate by the combinatorial possibilities for this reaction. For instance, for constructive reactions produced by pair-wise collisions, we can express it as:

$$a_f = k_f \cdot n(n-1). \tag{1}$$

At $t = \tau$, we sample a reaction based on the relative propensities, and this process is repeated until $t = t_{max}$. Although the original SSA provides precise results, it becomes computationally unfeasible when dealing with a large number of reactions. Consequently, we have employed the "τ-leaping" method, wherein the number of reactions occurring during an interval dt is approximated by a Poisson distribution with mean $a_i \cdot dt$?:

$$n_i \approx \mathcal{P}(a_i \cdot dt), \tag{2}$$

resulting in faster calculation times, with speed-ups ranging from one to two orders of magnitude.

In the logfiles, I've included a report of skipped reactions. For example,

```
Performed 10398531083 constructive reactions, skipped 0 (0.0 %)
Performed 398587084 destructive reactions, skipped 0 (0.0 %)
Performed 56930163 diffusion reactions, skipped 0 (0.0 %)
```

Having zero or next to zero skipped reactions is good. Having a high percentage—for example, more than 10%—means that dt is too large. Obviously the simulations at risk of having more skipped reactions are the ones where e.g. inflow or diffusion are highest.

For now, I've mostly used $dt = 10^{-3}$ and it seems to work pretty well. No skipped reactions even in high inflow/high diffusion regimes. Increasing to $dt = 10^{-2}$ remains stable, but doesn't provide much of a speed improvement (to be investigated later). However, increasing up to $dt = 10^{-1}$ yields a dramatic increase in skipped reactions.

4.2 MS08: distance from source (bis)

Reimplementation of the "distance from source" analysis, this time using the tau-leaping method and R to plot the figures. Nothing really new here, besides the inset which is slightly different from the one I was able to get with Julia. These are "density plots" though, so it implies a reconstruction of the (presumed) original distribution, which is why they might slightly differ between languages/implementations. Otherwise the fact that I'm able to reproduce the initial figure is a good thing, this validates not only my initial results but also the accuracy of the new (tau-leaping) implementation. {subsec:MS08}

In Figure 11, data points indicate the mean assembly index for each reactor, as a function of the distance from the source for that specific reactor. The three curves represent three diffusion regimes (low-, intermediate-, high-diffusion). The grey dashed box indicates that for several reactors of the same regime/distance from the source we can nonetheless get different mean assembly indices—prompting the question as to what determines this mean, besides the (topological property) of distance from the source.

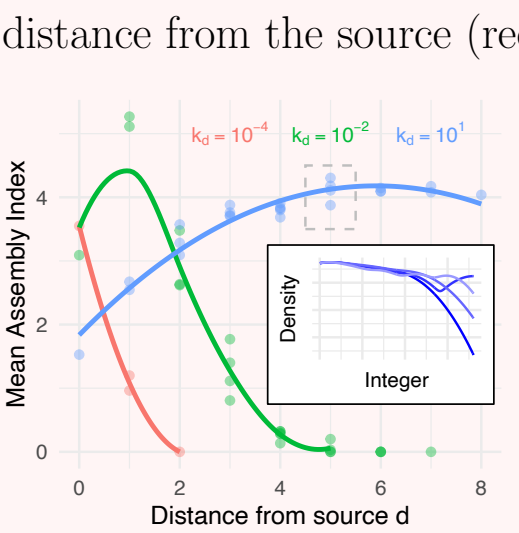

Figure 11: Mean assembly index for each reactor as a function of its distance from the source. Each data point corresponds to a reactor, with curves representing three distinct diffusion regimes (low, intermediate, high). The grey dashed box highlights that reactors equidistant from the source within the same diffusion regime can exhibit differing mean assembly indices—raising the question of what additional factors, beyond topological distance, influence the mean assembly index. {fig:ms08}

4.3 MS09: varying diffusion (bis)

{subsec:MS09}

Similar to the previous milestone: reimplementation of a previous analysis (MS02) that used the SSA algorithm, this time with the tau-leaping scheme and R to plot the figures. Figure 12 below is to be compared to 6: panel A shows the average populations varying across diffusion rates while panel B reports their standard deviation. Integers 2–9 are shown.

One notable difference between Figure 12 shown here and the previous Figure 6 is that for lower values of the diffusion rate population statistics differ significantly. These simulations were carried out some time ago as of the time I'm writing this, but to the best of my recollection we were confident that results in Fig. 6 must have been partly wrong somehow. If we interpret the curves in Fig. 12 as an indication of the three different regimes owing to the diffusion rate (fully mixed/heterogeneous/fully mixed again) the figure makes perfect sense.

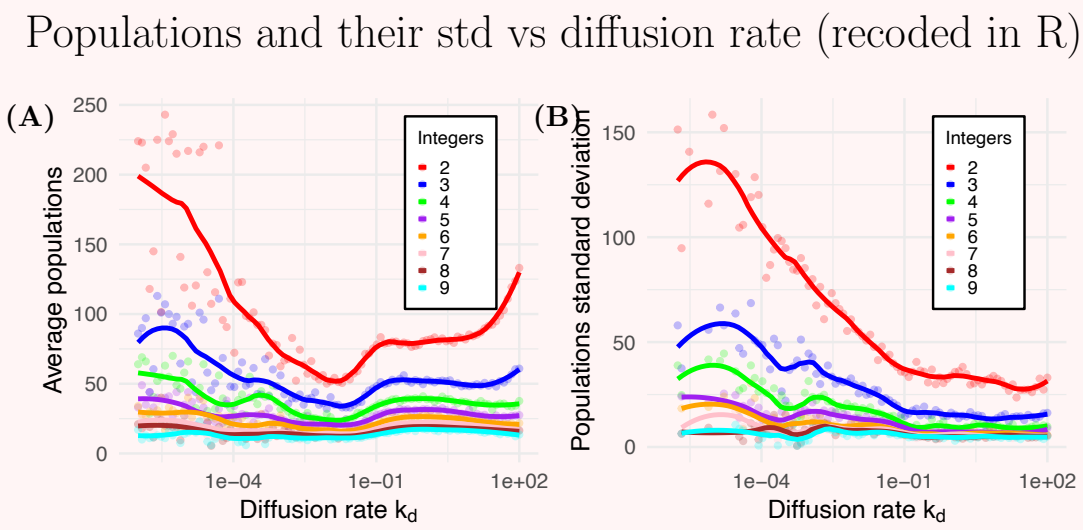

Figure 12: **(A)** average populations for integers 2–9. **(B)** standard deviation.

{fig:MS09-pop-std}

Following the same process, we have also carried out a reanalysis of the detection thresholds calculated previously. Again this is the same statistical ensemble of simulations, same parameters, etc. Figure 13 below is also quite identical to the one before (Figure 9).

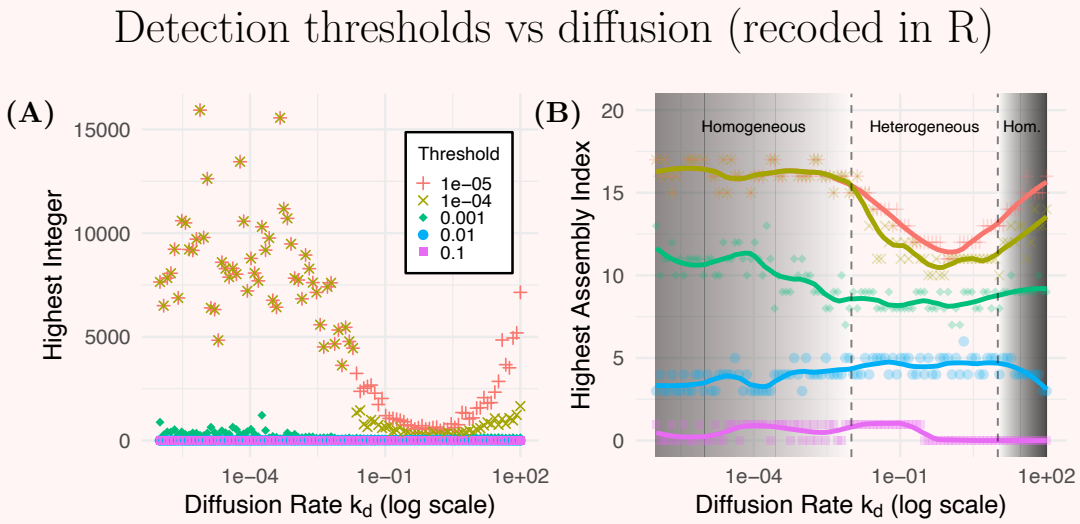

Figure 13: Reanalysis of the detection thresholds. Both panels are almost identical as the ones resulting from the previous analysis.

{fig:MS09-detection-thresholds}

4.4 MS12: assembly of integers (bis)

Same reanalysis of the relationship between integer and assembly index than before. Both figures are identical. {subsec:MS12}

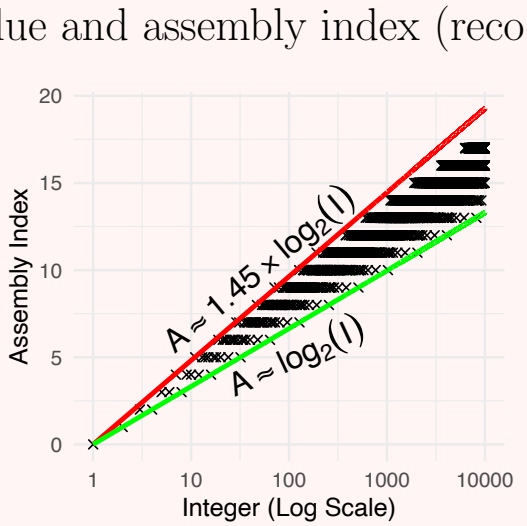

Figure 14: Reanalysis of the relationship between integer and assembly index. {fig:MS12-integer-assembly}

4.5 MS13: integers histogram

Show below on Figure 15 is the frequency distribution of molecules for a given simulation, at the final time step. The populations were taken over the whole system. The distribution is shown both as a function of integer value and assembly index. It's also truncated at integers of 15, as it extends much farther than that. {subsec:MS13}

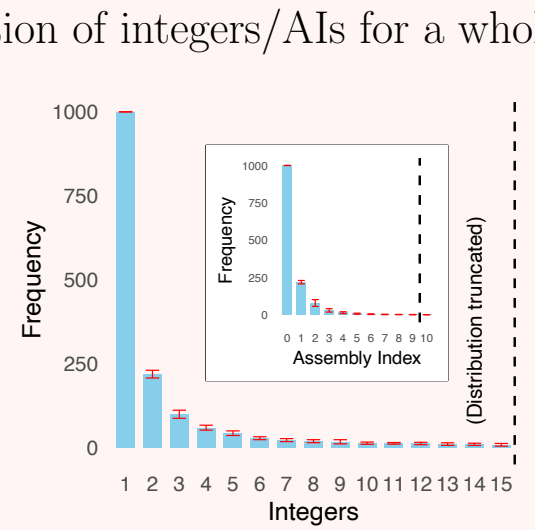

Figure 15: Distribution of molecules at the final time step, over the whole system. {fig:MS13-integer-histogram}

4.6 MS15-18: sweep on inflow × diffusion with $t_{\max} = 10^2, 10^3, 10^4$

{subsec:MS15-18}

Now we're ready to carry out extensive parameter sweeps.

Preliminary remark #1: The system's configuration is governed by several parameters: graph topology, number of inflows, reactor count, and the rates of destruction, construction, and diffusion. For the initial analysis we fix a square lattice with a single inflow and set the number of reactors to 25. We also fix the destructive rate at 1 to normalize the remaining reaction rates. This leaves three free parameters: the constructive rate, the diffusion coefficient, and the inflow rate. Because increasing the constructive rate predictably boosts molecular complexity, the present study concentrates on sweeping the inflow rate and the diffusion coefficient.

Preliminary remark #2: In this project we *do not* map our dimensionless parameters onto specific geochemical quantities. The model represents an idealized nanoporous medium in which chemistry unfolds, and we deliberately avoid committing to any particular origin-of-life scenario or entering detailed debates with geochemists.

Our initial exploration employed a Monte Carlo approach: we sampled the diffusion coefficient uniformly from $[10^{-3}, 10^{-1}]$ and the inflow rate from $[10^3, 10^6]$, executed 100 simulations, and plotted the results in the corresponding two-dimensional parameter space, with point colours reflecting the average integer complexity. With only 100 runs, the sampling was too sparse to reveal clear trends, and time-series plots exposed rich, non-trivial species dynamics. Consequently, I shifted to a more systematic study using the large multi-panel plots that proved successful in previous projects.

4.6.1 Time series for a single simulation and for whole systems

Figure 16 presents additional results. We ran an ensemble of 100 simulations on a 5×5 lattice, with columns corresponding to three run times, $t_{\max} = 10^2, 10^3, 10^4$ iterations. The time step was fixed at $dt = 10^{-3}$ to avoid skipped reactions; larger steps (10^{-2} or 10^{-1}) skipped anywhere from a few to several dozen percent of reactions while yielding only a modest 40–60 % speed-up.

The top row shows the time series from a representative simulation. Only the longest runs ($t_{\max} = 10^4$) visibly attain steady state.

The second row plots the lattice-wide populations of the first few species. Panels are ordered from low to high diffusion (left to right) and from low to high inflow (bottom to top). As we will see, these traces offer limited additional insight.

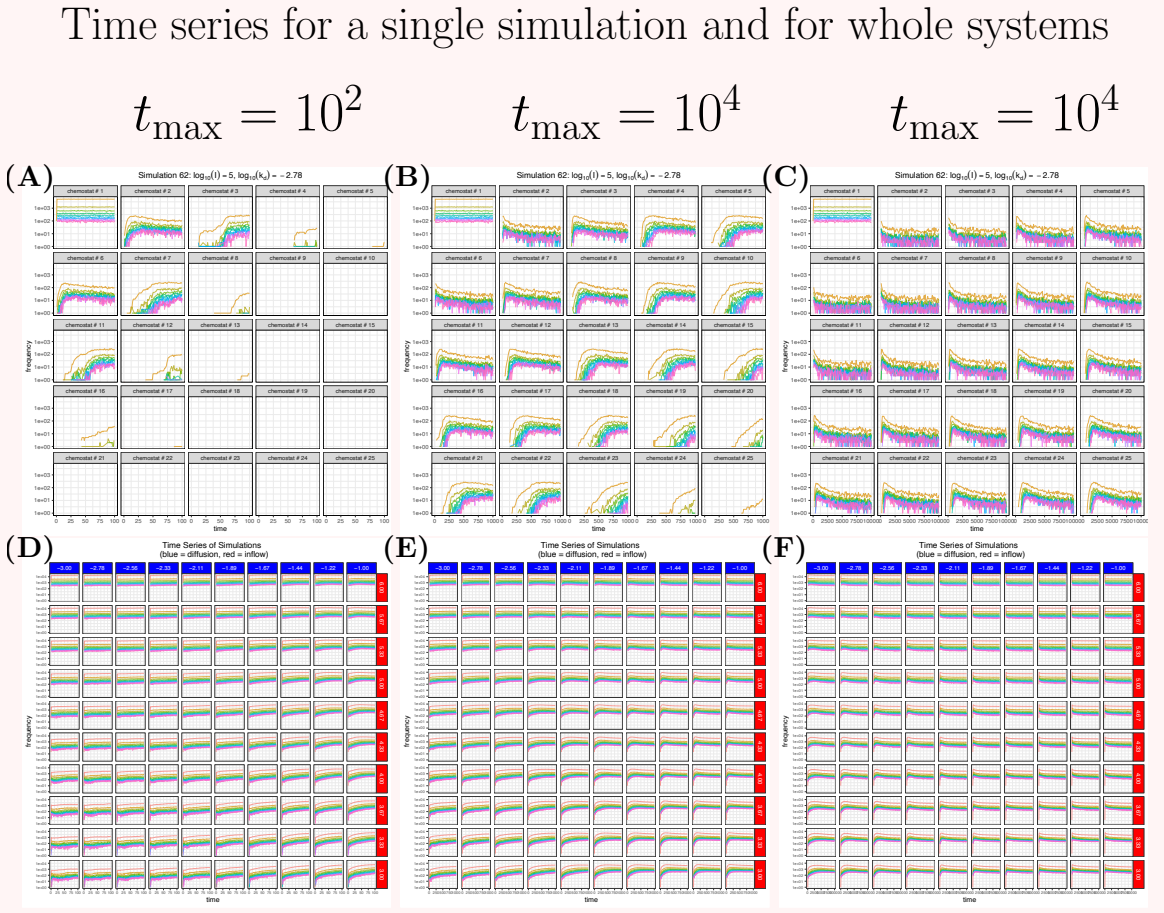

Figure 16: Time-series diagnostics for the lattice ensemble. Each column corresponds to one of three simulation lengths, $t_{\max} \in 10^2, 10^3, 10^4$ iterations; all runs use a 5×5 lattice and a fixed step size $dt = 10^{-3}$. **Top row:** Trajectories from a representative simulation illustrate that macroscopic steady state is achieved only in the longest runs ($t_{\max} = 10^4$). **Bottom row:** Lattice-averaged populations for the first few species, with panels ordered from low to high diffusion (left → right) and low to high inflow (bottom → top). These global averages reveal limited structure compared with the full spatially resolved data discussed later.

{fig:MS15-18-time-series}

4.6.2 Power law fit over the whole system (integer/AI) and inflow/outflow

Our next results are presented in Fig. 17. We continue to work with the same dataset but now offer additional views.

The first row displays probability–density functions for the first 50 chemical species at the final time step, plotted as scatter points in log–log space. The red line shows a power-law fit performed in log space. Agreement is good for these initial species, yet it degrades once we broaden the sample (e.g., to 200 + species), revealing an apparent heavy-tailed distribution that merits further scrutiny. The fitted law follows A^{-k}, and the dashed line indicates a slope of $k = 1$—a slope below 1 would imply a distribution with an undefined mean.

The second row repeats the analysis with the calculated assembly index replacing the integer species label. This panel remains provisional: the logarithmic binning may be sub-optimal, and the assembly-index calculation still needs validation. Because the assembly index scales roughly with the logarithm of the integer label, plotting it on a log axis effectively applies a double-log transformation, whose appropriateness we must evaluate.

The third row is the most revealing. By separating the inflow and outflow reactors, it lets us quantify the topological influence between them. A clear trend emerges: the outflow reactor (orange points, red fit) has a noticeably shallower slope than the inflow reactor (green points, blue fit), suggesting the downstream formation of more complex molecules. Before drawing firm conclusions, however, we must first normalise the PDFs—a task we will revisit.

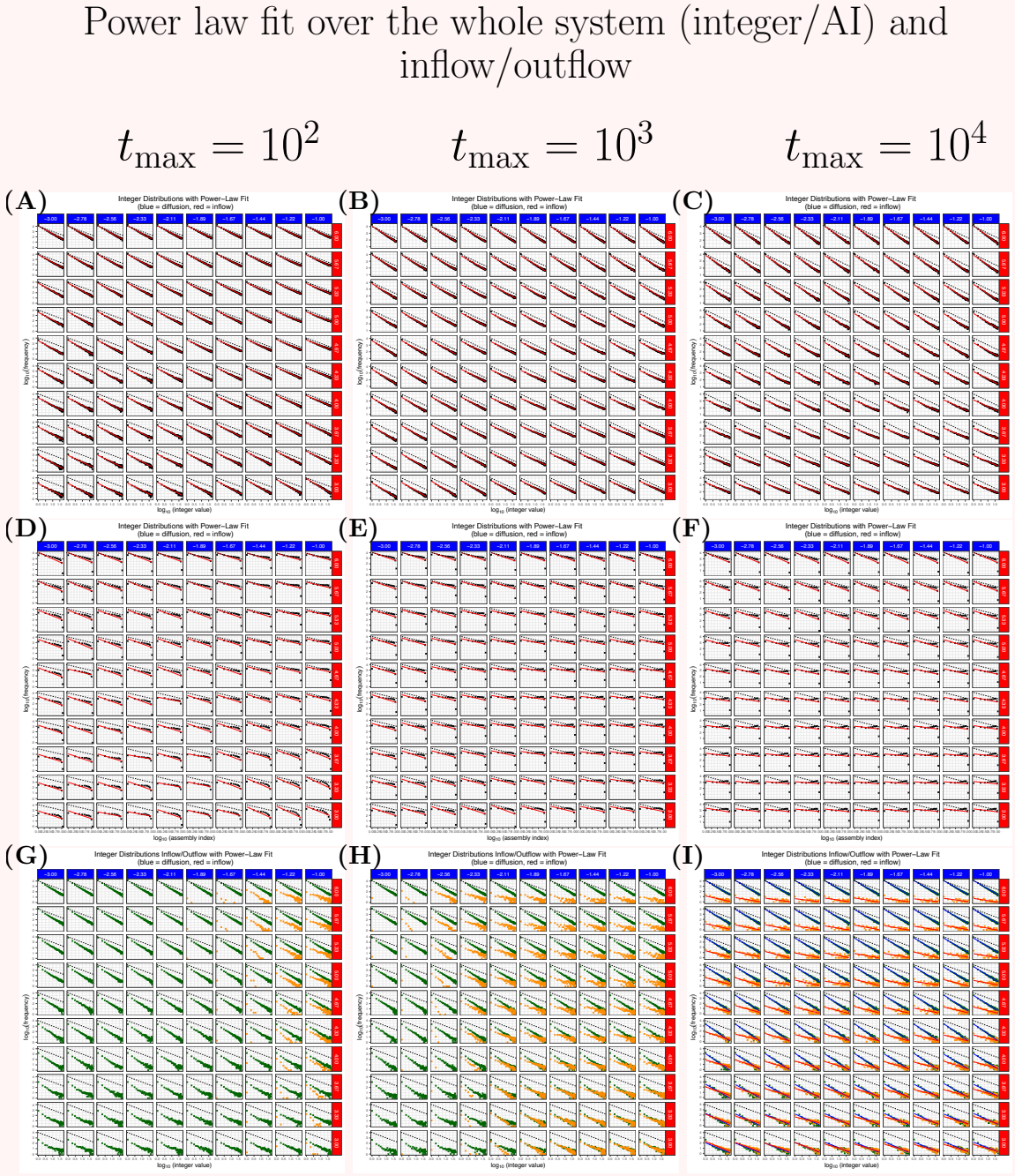

Figure 17: **Top**: PDFs of the first 50 species in log–log space; red line is the power-law fit, dashed black denotes slope $k = 1$. **Middle**: Same data plotted against assembly index (AI); binning and double-log scaling still under review. **Bottom**: Inflow (green, blue fit) vs. outflow (orange, red fit) reactors—shallower outflow slope hints at greater downstream complexity. Normalisation of PDFs is pending.

{fig:MS15-18-power-law-fit}

4.6.3 Exponent of power law fit, average integer and total mass

Figure 18 shows a series of heatmaps built from the same dataset.

First row — power-law exponent. In the third column, where simulations have reached steady state, diffusion (horizontal axis) exerts almost no influence, whereas inflow (vertical axis) strongly shapes molecular complexity. Higher inflow raises the exponent, steepening the slope and reducing the abundance of complex species. Note, however, that this map uses the exponent fitted over the entire system; later fits partitioned by inflow/outflow are more informative.

Second row — average integer. Averaged across the system, diffusion again has negligible effect, while greater inflow monotonically increases the mean integer value.

Third row — total mass. The total mass pattern closely mirrors that of the average integer, reinforcing the dominant role of inflow over diffusion in these simulations.

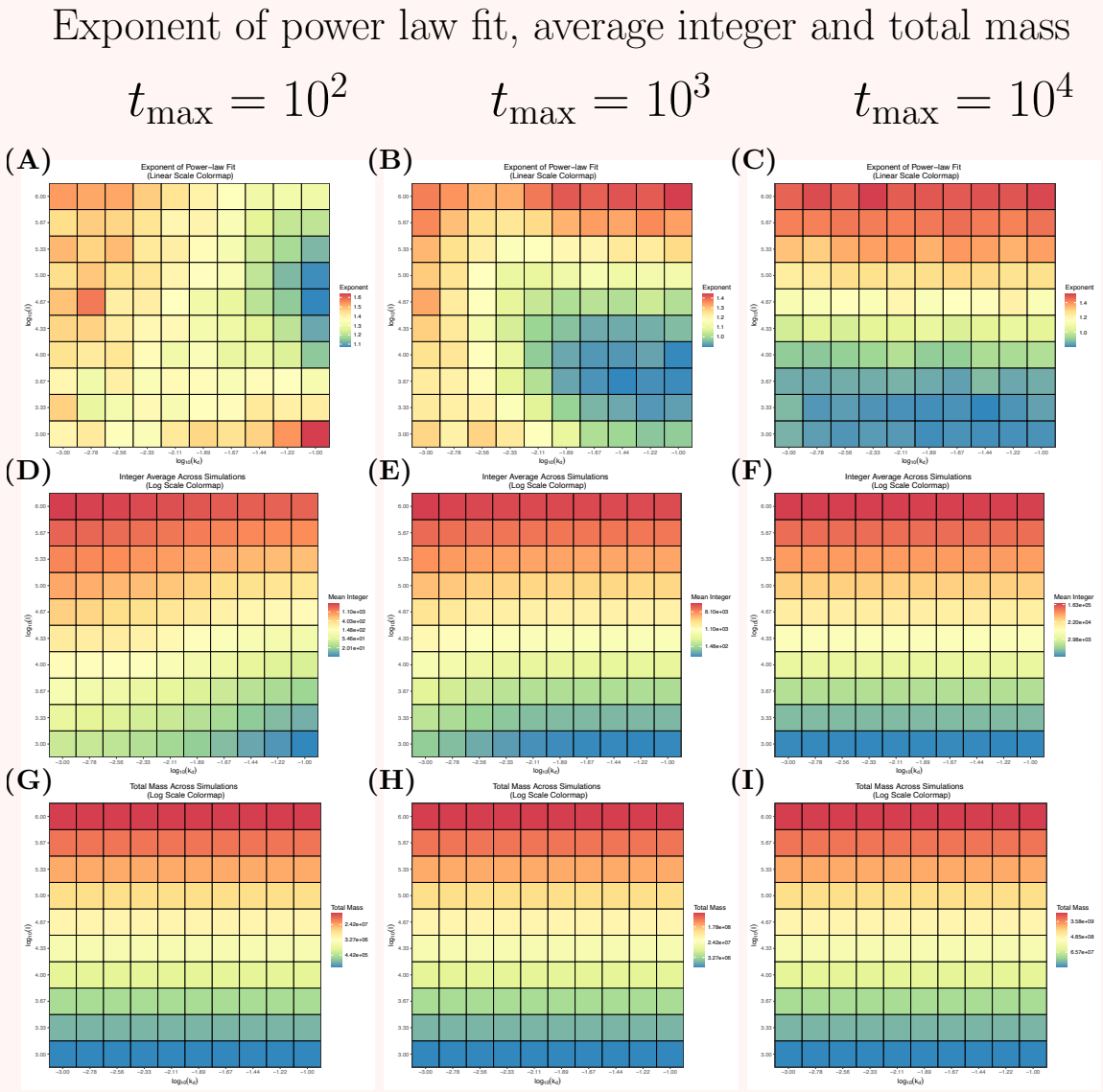

Figure 18: Heatmaps across diffusion rate k_d (horizontal axis) and inflow I (vertical axis). Row 1: power-law exponent of the species-abundance distribution—larger I steepens the slope, whereas k_d has little effect. Row 2: mean integer index, rising with I and insensitive to k_d. Row 3: total molecular mass, mirroring the mean-integer trend. All panels derive from the same dataset; steady-state results appear in the third column.

{fig:MS15-18-exponent-int-mass}

4.6.4 Exponents of power law fit at inflow/outflow/diff

Figure 19 displays three heatmaps of the power-law exponents for the inflow and outflow reactors, using an adjusted color scale to accentuate subtle simulation-to-simulation differences. Panel A plots the inflow exponent k_{inflow}, Panel B the outflow exponent k_{outflow}, and Panel C their difference $k_{\text{outflow}} - k_{\text{inflow}}$.

In Panel A, diffusion again exerts virtually no influence, whereas inflow dominates: the exponent is lowest at small inflow, rises sharply with increasing inflow, then settles to an intermediate plateau.

Panel B shows a similar absence of diffusion effects, but the inflow dependence is muted—k_{outflow} decreases only slightly as inflow grows.

Consequently, the difference map in Panel C largely mirrors the inflow pattern of Panel A; the contrast arises almost entirely from changes at the inflow reactor.

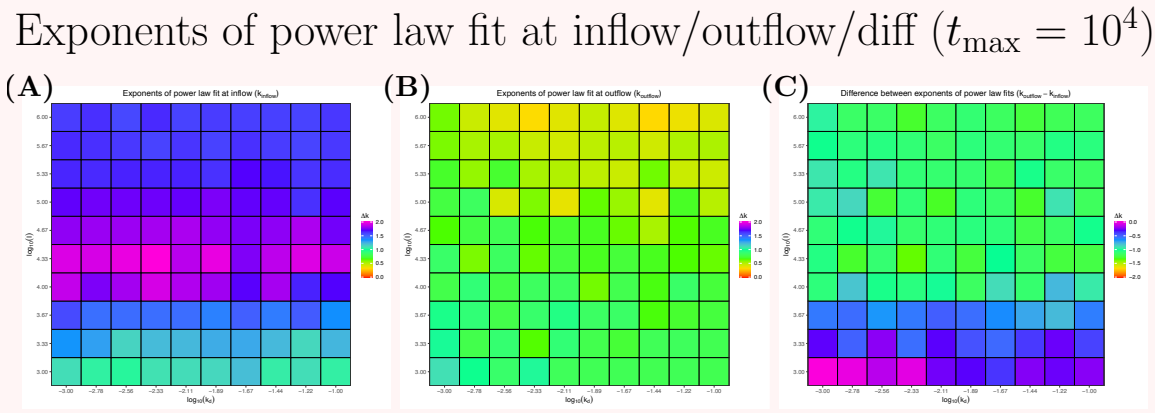

Figure 19: **Power-law exponents at inflow and outflow.** Heatmaps plot the exponent of the power-law fit for each reactor as a function of diffusion (x-axis) and inflow rate (y-axis). **A)** Inflow exponent k_{inflow}: diffusion has negligible influence, while increasing inflow first raises the exponent sharply and then levels it off. **B)** Outflow exponent k_{outflow}: diffusion again has no discernible effect, and the inflow dependence is modest, with a slight decrease at higher inflow. **C)** Difference $k_{\text{outflow}} - k_{\text{inflow}}$: the pattern closely tracks panel A, confirming that variations at the inflow reactor drive the contrast between reactors.

{fig:MS15-18-exp-inflow-outflow}

4.6.5 Time series for limit cases

Figure 20 displays the complete time-series traces for all reactors in the four "limit-case" simulations—those using the extreme combinations of diffusion and inflow parameters:

(A) $\log k_d = -3$, $\log I = 6$

(B) $\log k_d = -1$, $\log I = 6$

(C) $\log k_d = -3$, $\log I = 3$

(D) $\log k_d = -1$, $\log I = 3$

Beyond illustrating that each system has effectively reached steady state, the plots reveal no further noteworthy differences.

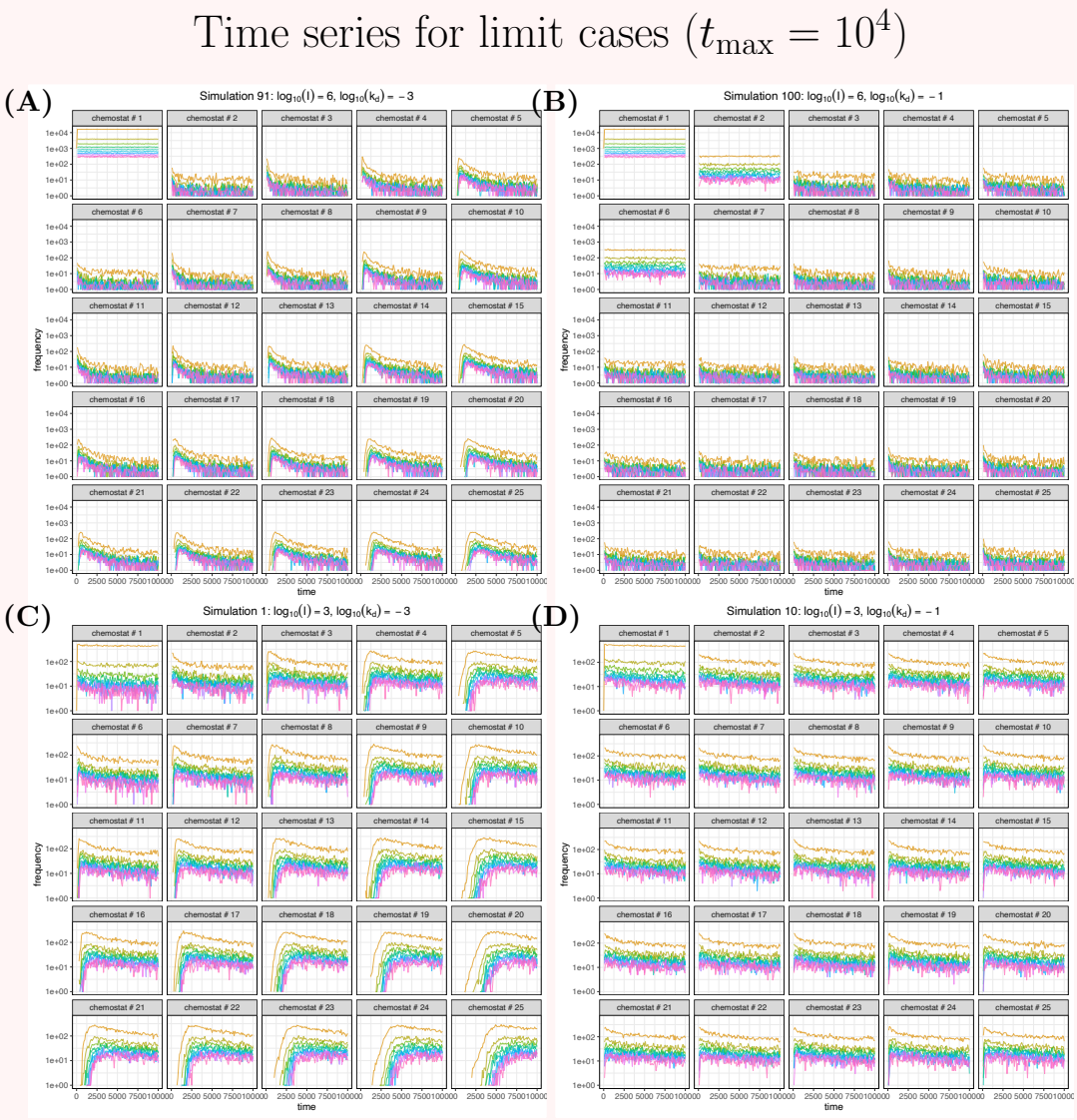

Figure 20: Time-series profiles for every reactor in the four "limit-case" simulations, defined by the extreme combinations of diffusion ($\log k_d$) and inflow ($\log I$) parameters. All traces level off, indicating that each system has effectively reached steady state.

{fig:MS15-18-limit-cases}

4.6.6 Wrapping up MS15-18

{subsubsec:wrap-up-15-18}

We have produced numerous figures, not all equally informative. It is therefore time to consolidate our results and retain only those that best support our analysis. Figure 40 presents the key panels identified so far: the limit-case simulations, the power-law fits for inflow and outflow reactors, heatmaps of the corresponding exponents, and the plots of average integer and total mass.

In the next section, we will compare these findings across alternative topologies to assess how network structure influences the emergence of complex molecules.

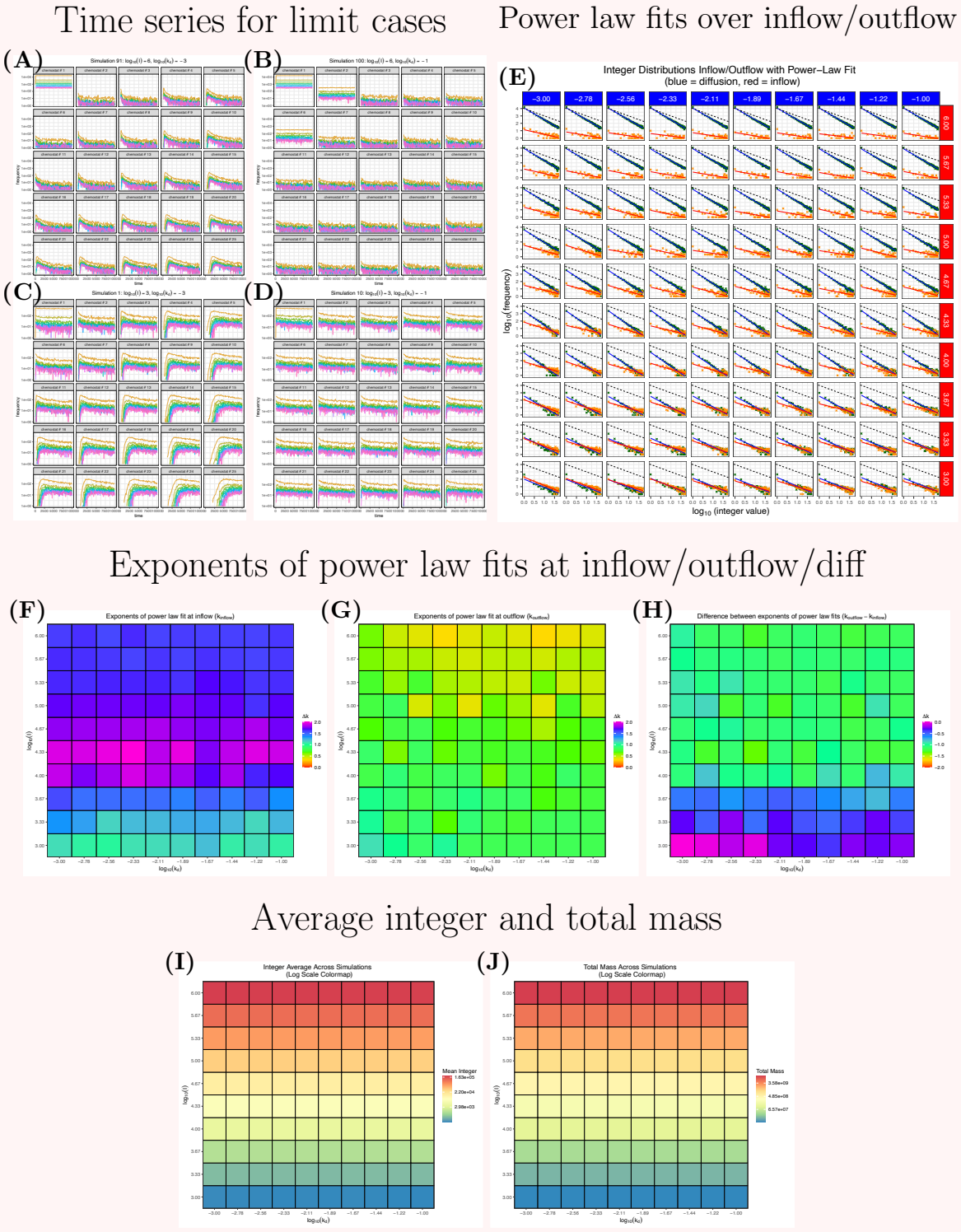

Figure 21: Key outputs: limit cases, power law fits and exponents, average integer and total mass.

{fig:wrap-up-15-18}

4.7 MS19: randomizing edges of the lattice topology

{subsec:MS19}

Next we'll be repeating the previous analysis, this time using a lattice where we randomized edges. We use a directed edge-swapping algorithm to randomize the topology of a graph while preserving each node's in-degree and out-degree. The procedure is as follows:

1. Randomly select two directed edges, $(u_1 \to v_1)$ and $(u_2 \to v_2)$, such that all four nodes are distinct.
2. Propose swapping the target nodes to form new edges $(u_1 \to v_2)$ and $(u_2 \to v_1)$.
3. Check that the new edges do not introduce self-loops or duplicate edges.
4. If valid, perform the swap by deleting the original edges and adding the new ones.
5. Repeat for a fixed number of swaps, typically 10 times the total number of edges, to sufficiently randomize the network structure.

This process produces a randomized null model of the original network that preserves the full in-degree and out-degree sequence. For each simulation of the current milestone, we will apply this algorithm to the lattice topology, creating a new randomized version (i.e., 100 different randomized graphs for the 100 simulations of the ensemble).

Oops. This analysis is on hold as there was a bug in the randomization of the network. Pending discussion with Cole on this.

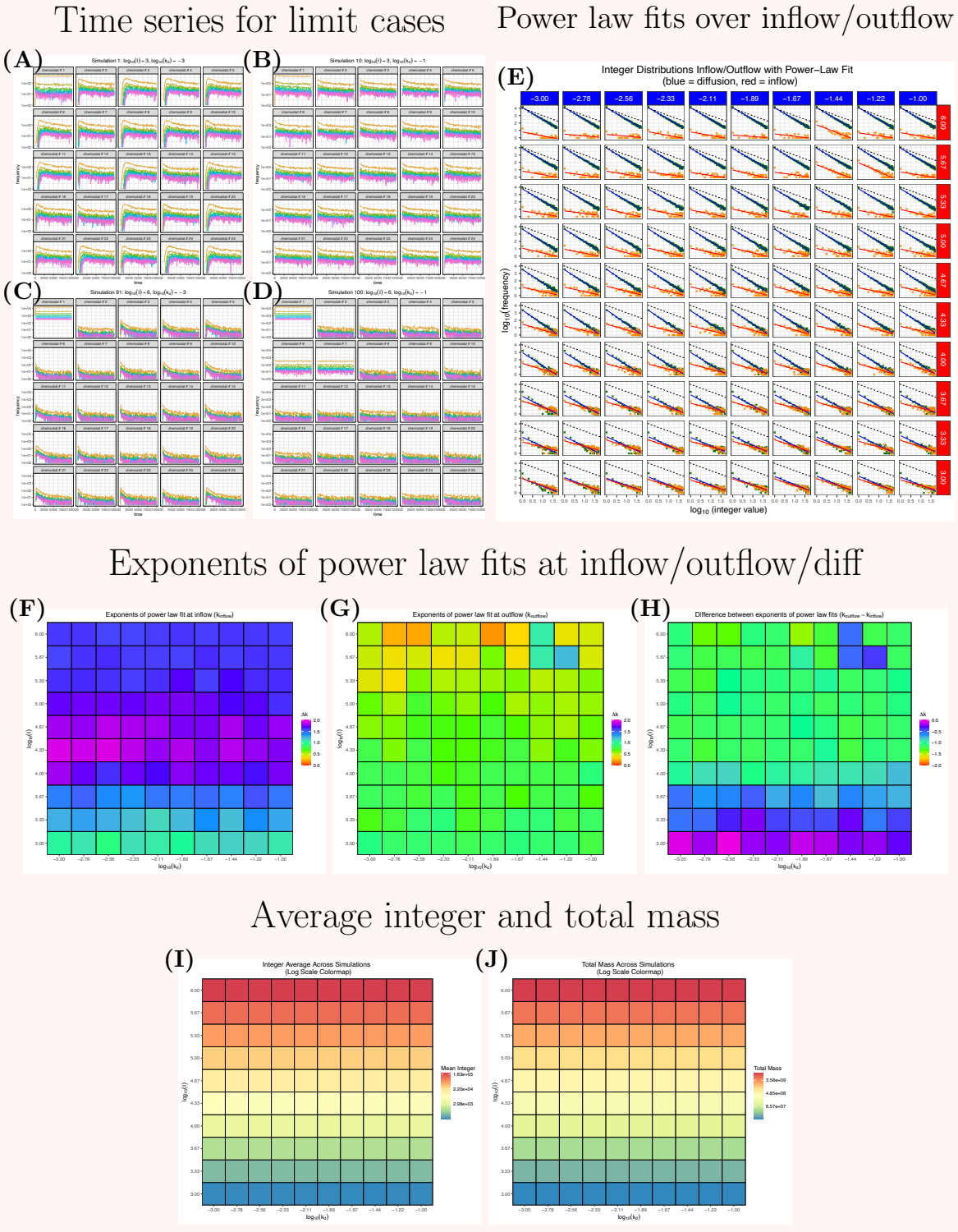

Figure 22: Key outputs for the randomized version of the lattice: limit cases, power law fits and exponents, average integer and total mass.

{fig:wrap-up-15-18}

4.7.1 Comparing lattice vs randomized topologies

Since the heatmap for the outflow reactor provides the most informative view of the data, we now compare this heatmap for the lattice topology with that of its randomized counterpart.

Oops. This analysis is on hold as there was a bug in the randomization of the network. Pending discussion with Cole on this.

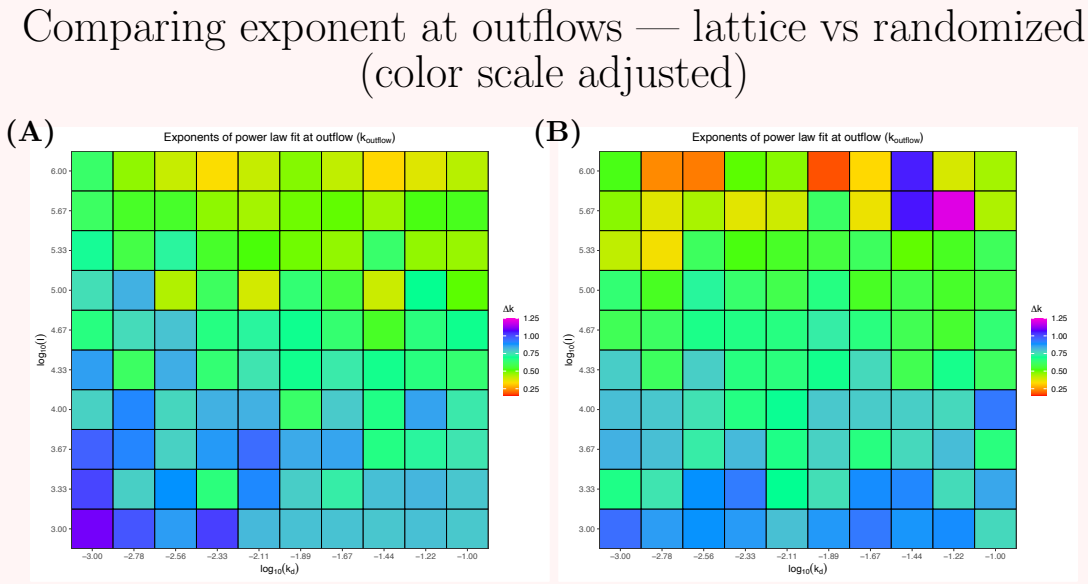

Figure 23: **(A)** ...

{fig:ms20}

5 Experiments with the tau-leaping method (with the outflow fixed)

Okay so the bug in the network's randomization (just discussed in the previous section) was fairy minor. Basically the randomization had no effect. But upon fixing that, we could clearly see something was not right, as the outcome of the simulations didn't change much.

Turns out there was an even more serious bug: the outflow of the network was defined implicitely as the reactor that didn't have any exiting edges (i.e., edges from this reactor to another one). The algorithm, seeing no exiting edge, just removed molecules while applying diffusion but didn't add these to any other reactor. That was the implicit outflow.

However, a few months ago we decided that diffusion should work both ways between reactors, instead of one-way. We therefore added additional edges pointing to the opposite direction to every existing edge in our networks, which was the simplest solution to this. However this had the consequence of creating outgoing edges for the last reactor (the outflow). The molecules consequently accumulated in the system, never exiting.

The solution to this was to completely de-couple the diffusion and the outflow. I've created a new `outflow_rate` parameter (that can optionally be set to match the diffusion rate) and the outflow sub-algorithm doesn't depend anymore on the presence of outgoing edges. For now we manually define the outflow reactor. We'll improve this when dealing with more complex topologies than lattices.

All of this however means that I have to re-calculate a bunch of simulations. We'll be re-calculating from MS08 (the distance-from-source figure), MS09 (diffusion sweeps and detection thresholds) and then we'll move on to the inflow+diffusion sweeps in MS15-MS18.

One last detail: a while ago we were using an implicit inflow as well, keeping the mass in the first (inflow) reactor fixed to e.g. 1000 molecules and relying on the diffusion to move these throughout the system as we replenished the reactor. We switched eventually to having an inflow defined in molecules/unit time. We'll be keeping this in what follows. This means the new simulations won't compare directly to the previous ones, but that shouldn't make that much of a difference.

5.1 MS23: distance from source (re-bis)

First result from our re-calculation: the distance from source figure (MS08). These new simulations were calculated using the tau-leaping method using $dt = 10^{-3}$ (there was no skipped reaction whatsoever, even at very high diffusion). We initially integrated over $t_{max} = 100$ but had to extend this to $t_{max} = 10^5$ to make sure the low diffusion sims reached steady state. We again used a lattice topology (there is no randomization here) with $N = 25$ reactors. The inflow was set to $I = 10^3$ molecules/second, the forward rate to $k_f = 10^{-3}$ and the diffusion rate sampled in $\log k_d \in [-6, 1]$. The outflow rate was set to be equal to the diffusion rate.

Figure 24 shows the time series for representative simulations over the range of diffusion rates that was sampled. Simulations at $\log k_d > -4$ seem to be at (or at least, very close to) steady-state. The figure also shows the average assembly index per reactor, plotted against the distance of that reactor from the source reactor. Three curves display three diffusion regimes as before and the inset shows density plots for a few simulations (grey dashed box) where the average differs even though the distance to source is the same. Density plots are reconstructions of the PDF using kernel density estimates, carried out using ggplot's `geom_density()` function.

Overall these new results do not alter our initial conclusions. For a given distance-to-source, the average AI for a given reactor differ from one instance to another.

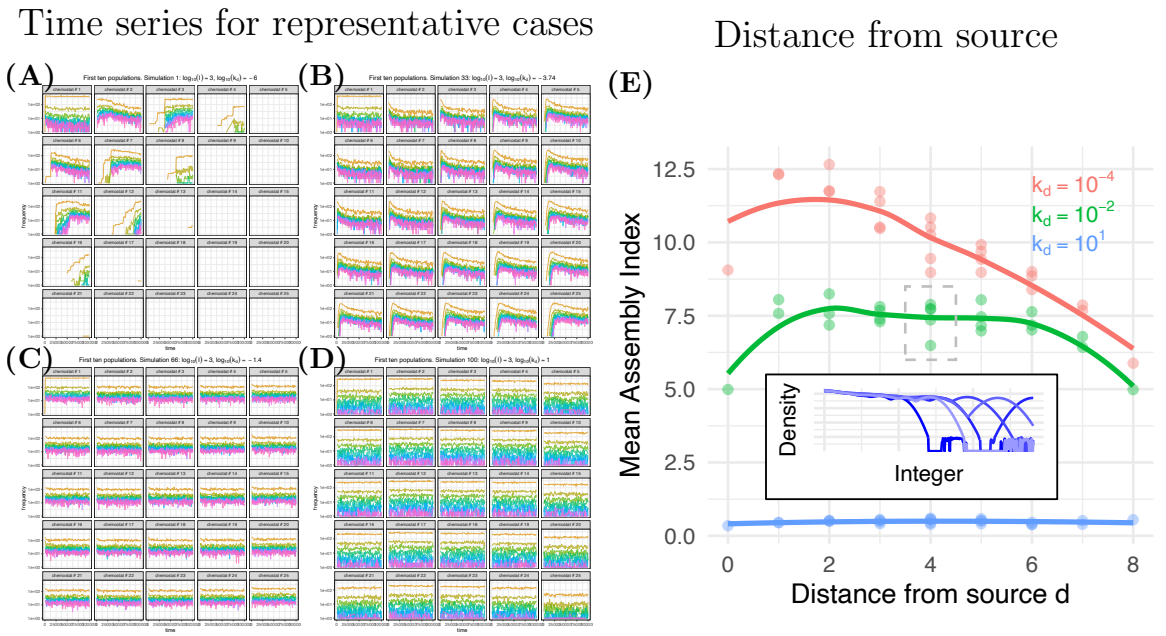

Figure 24: Time series for representative simulations, and distance from source plot for three diffusion regimes. See text for details.

{fig:MS23}

A few additional results: total number of molecules + total mass, per reactor. Shown below for a representative simulation near $\log k_d \sim -4$. This is relevant because we can look at reactor #25 and see that both of these pretty much cease to increase. This is another way to determine whether the simulation is at steady-state, and we're not limited to only seeing the first ten chemical species like on the previous figure. This would be especially relevant if the more complex species continue increasing while the simple stuff remains steady.

Total # molecules per reactor

Total mass per reactor

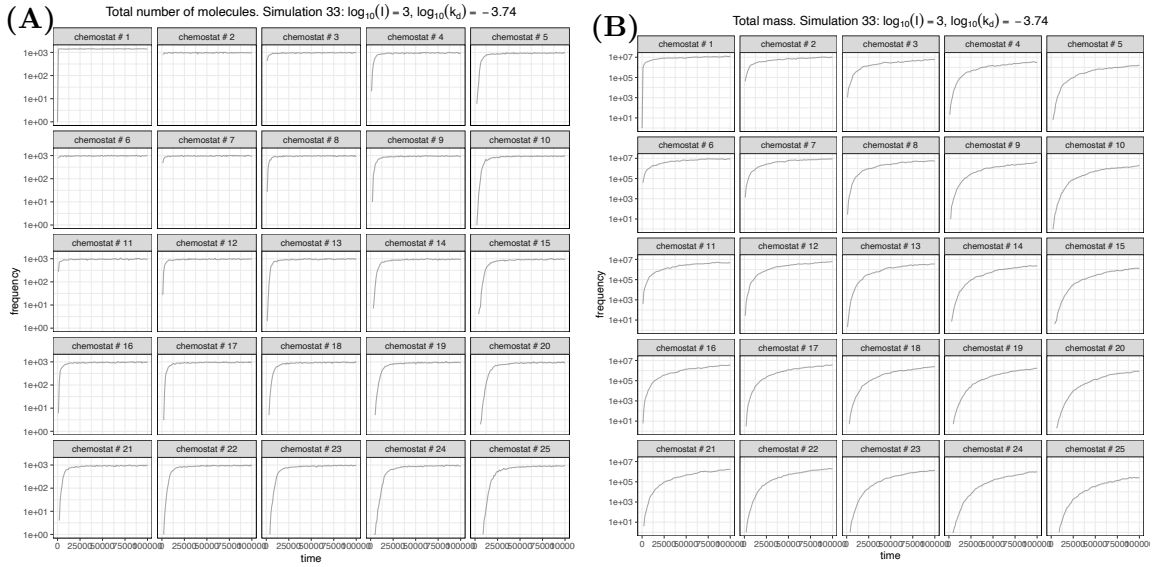

Figure 25: Number of molecules and total mass per reactor for a representative simulation near $\log k_d \sim -4$.

{fig:MS23b}

One last result: we'll compute the total Assembly as defined in (Sharma et al., 2023):

$$A = \sum_{i=1}^{N} e^{a_i} \left(\frac{n_i - 1}{N_{\mathrm{T}}} \right)$$

where we iterate on N *unique* objects in the system, with a_i being the assembly index of object i, n_i its copy number, N_T the total number of (non-unique) objects in the system. We can either calculate the total assembly A of a chemostat, or calculate it for the whole system altogether (A is not additive). The figure below shows both, including $A_\Delta \equiv A_{\mathrm{outflow}} - A_{\mathrm{inflow}}$.

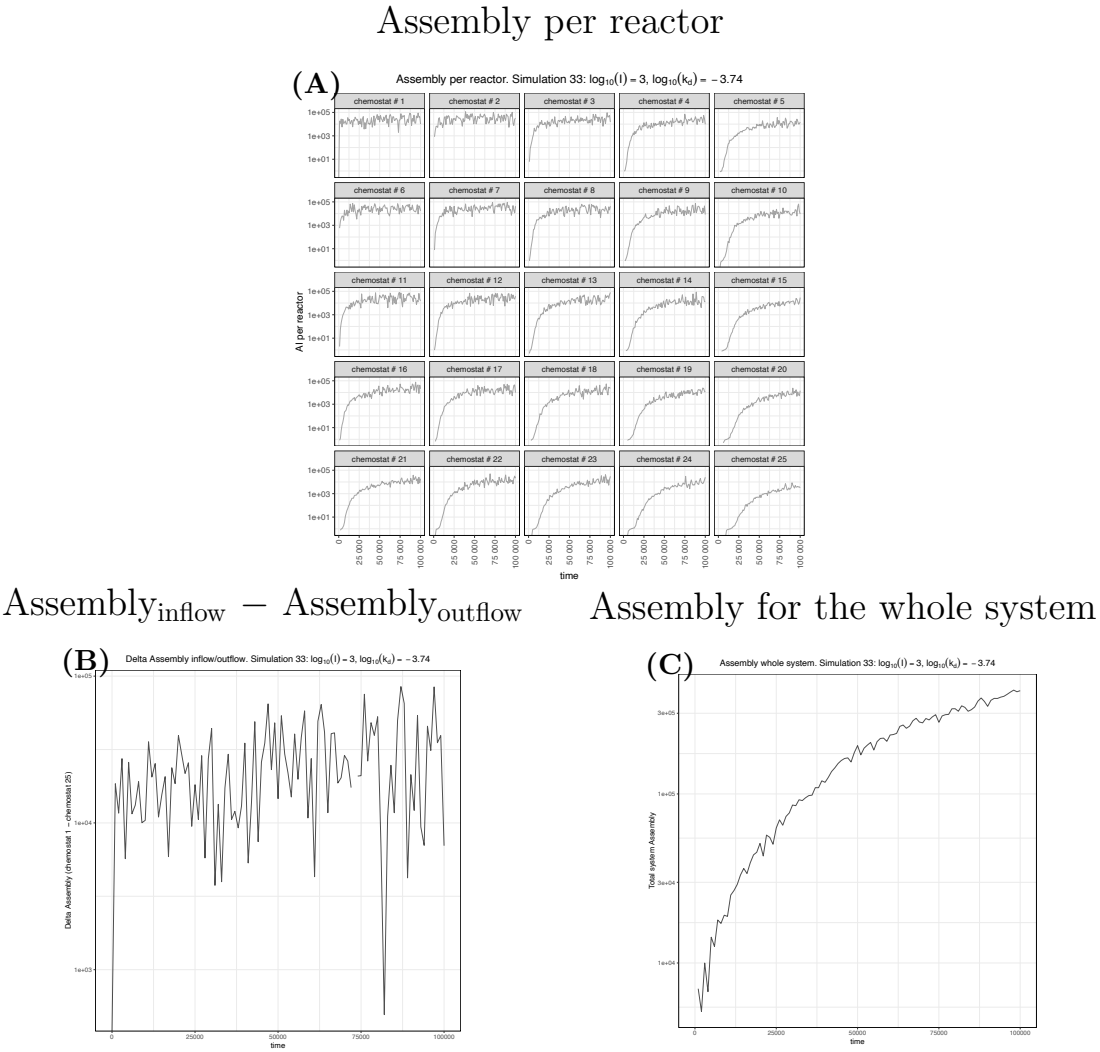

Figure 26: Assembly per reactor, delta outflow/inflow and assembly for the whole system, for a representative simulation near $\log k_d \sim -4$.

{fig:MS23c}

5.2 MS24: sweep over k_d

We're re-calculating here the figures from MS09: (mean) population variations over a diffusion parameter sweep, their standard deviation, and the most complex molecules we can detect for a given detection threshold. We're reusing the same simulations from the last subsection, integrated over 10^5 iterations. The population averages are calculated over the whole system (i.e. across chemostats). The detection thresholds take into account individual chemostats: if we're defining a threshold of X, we'll want to know what molecule has at least X^{-1} copies *in a given chemostat*—i.e. we're assuming that our detection tool can detect with a certain threshold for a single chemostat. Detection thresholds lower than 10^{-4} are not shown since they are equal to the latter (i.e., no molecule has a copy number higher than 10^4 in a given chemostat for these simulations). Representative simulations for $\log k_d \in \{-6, -5, \cdots, 0, 1\}$ are shown on the next page.

Panel A on the figure below is very interesting: we're "loading up" the system at $k_d < 10^{-4}$ (again, simulations aren't at steady state in this region) and then populations decrease as diffusion starts to impact the dynamics (around $k_d \sim 10^{-3}$). After that, populations increase again until diffusion gets very high.

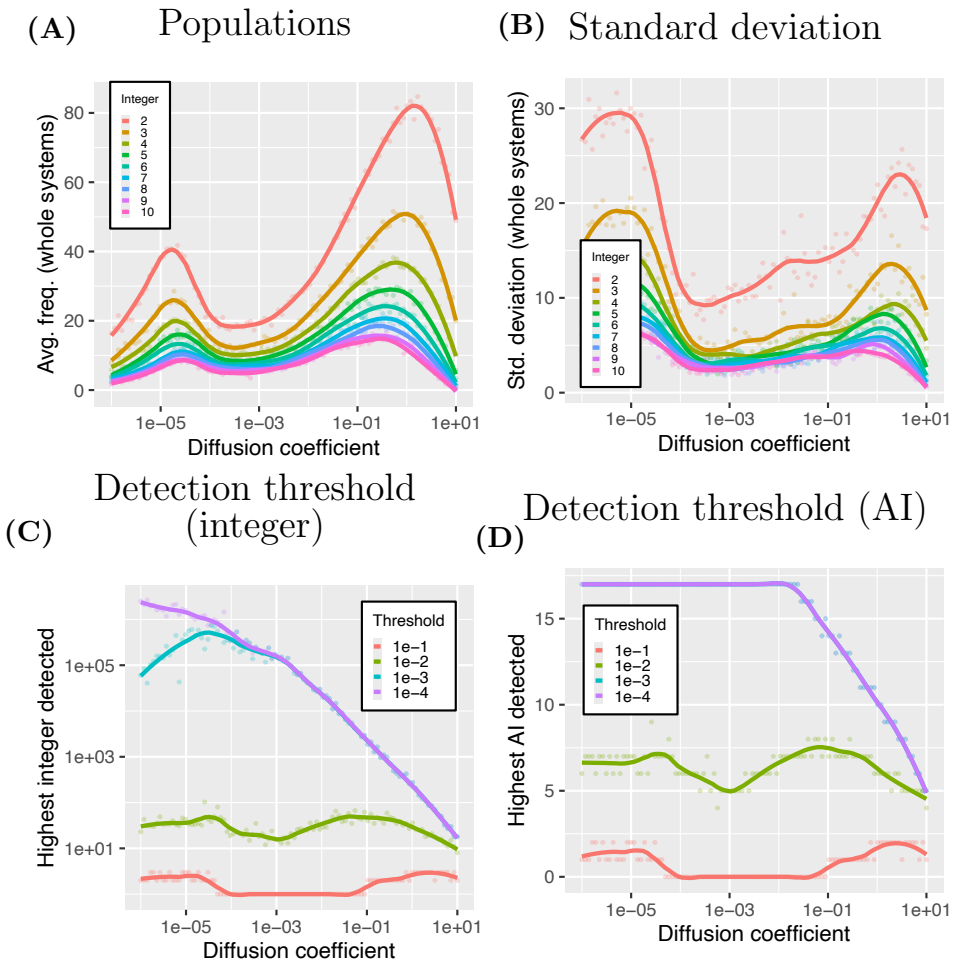

Figure 27: Average populations and their standard deviation, calculated across the whole system, as well as detection thresholds in terms both of integers and Assembly Index.

{fig:MS24b}

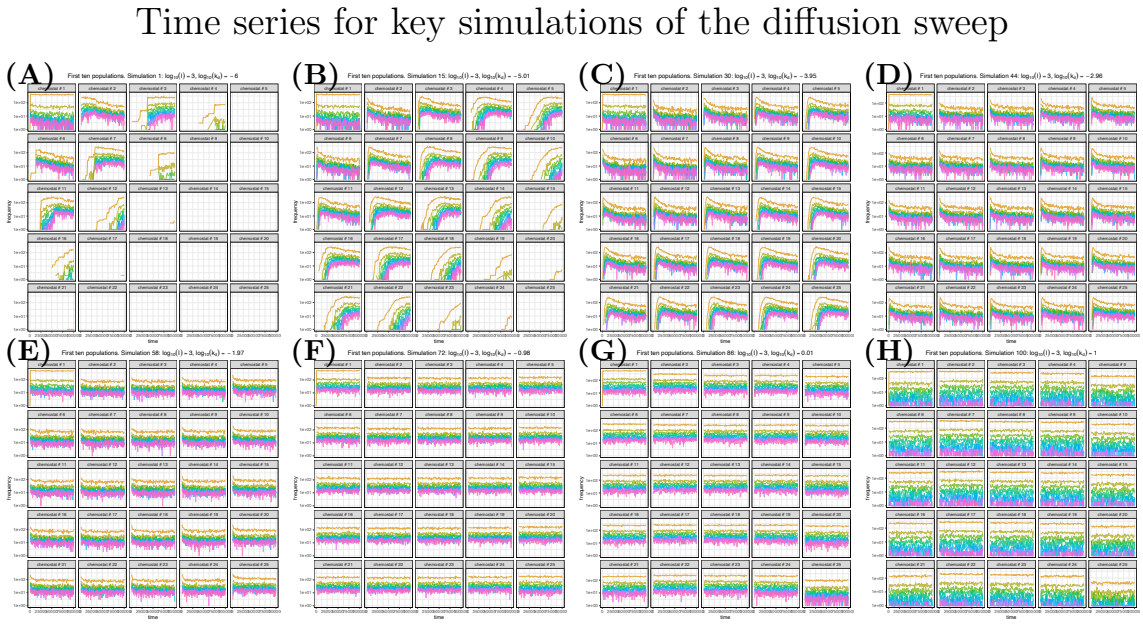

Figure 28: Time series for key simulations in the diffusion parameter sweep. The six panels show simulations where $\log k_d \in \{-6, -5, \cdots, 0, 1\}$.

{fig:MS24b}

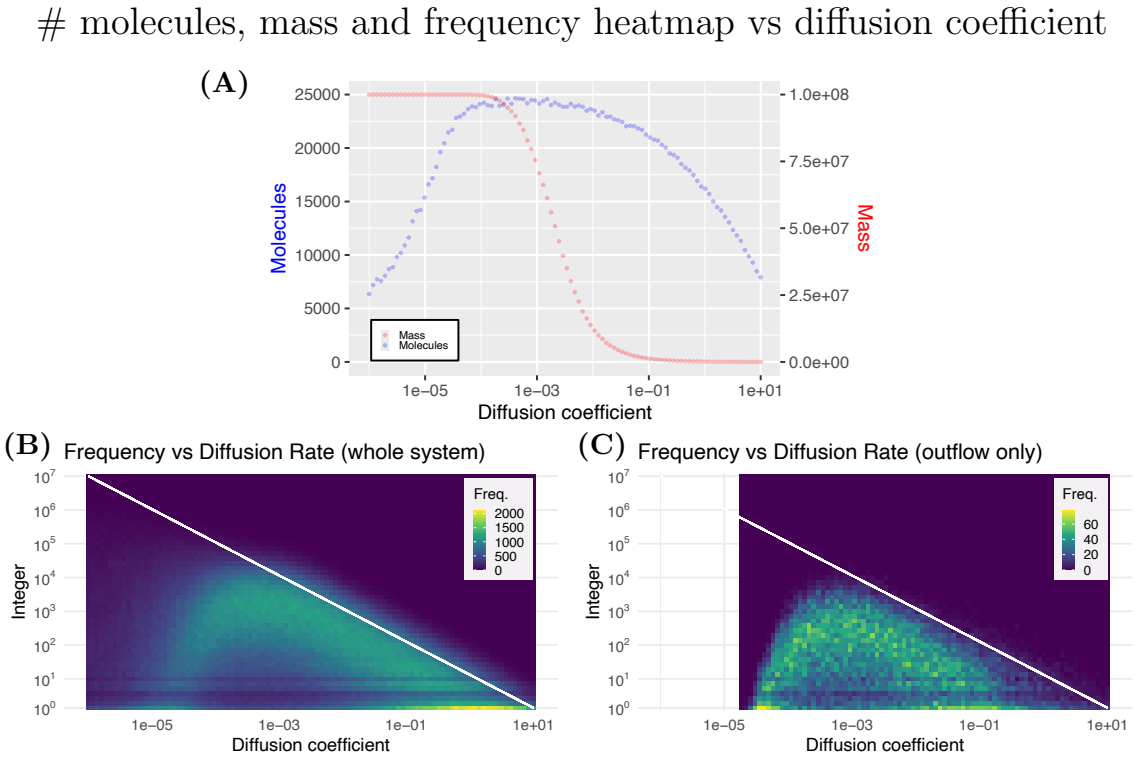

Figure 29: Panel A shows total number of molecules as well as total mass for whole systems across k_d's. Panels B & C show integer distribution frequencies for either whole systems or outflow reactors. White line guides the eye (slope ~ -1).

{fig:MS24c}

5.3 MS25: sweep over inflow × diffusion: benchmark

We want to do a parameter sweep, this time over both inflow and diffusion, which is a re-calculation of MS15-18. However while re-calculating things since we fixed the bug with the outflow, we noticed that several simulations used previously hadn't reached steady state. In our previous sweep over k_d, we integrated over 10^5 iterations, which made the simulations having $k_d > 10^{-4}$ reach steady state. Therefore, our first step will be to do a quick sweep over a large range of parameter values to determine in what regimes are the simulations reaching steady state. We'll try integrations over $10^3, 10^4$ and 10^5 time steps.

Zoom me (yes this is hacky): integration over 10^3 time steps
low resolution shown, original figure too heavy

Figure 30: All chemostats for simulations of the benchmark. Moving rightward increases the inflow, moving downwards increases diffusion.

{fig:MS25a}

Integration over 10^4 time steps
low resolution shown, original figure too heavy

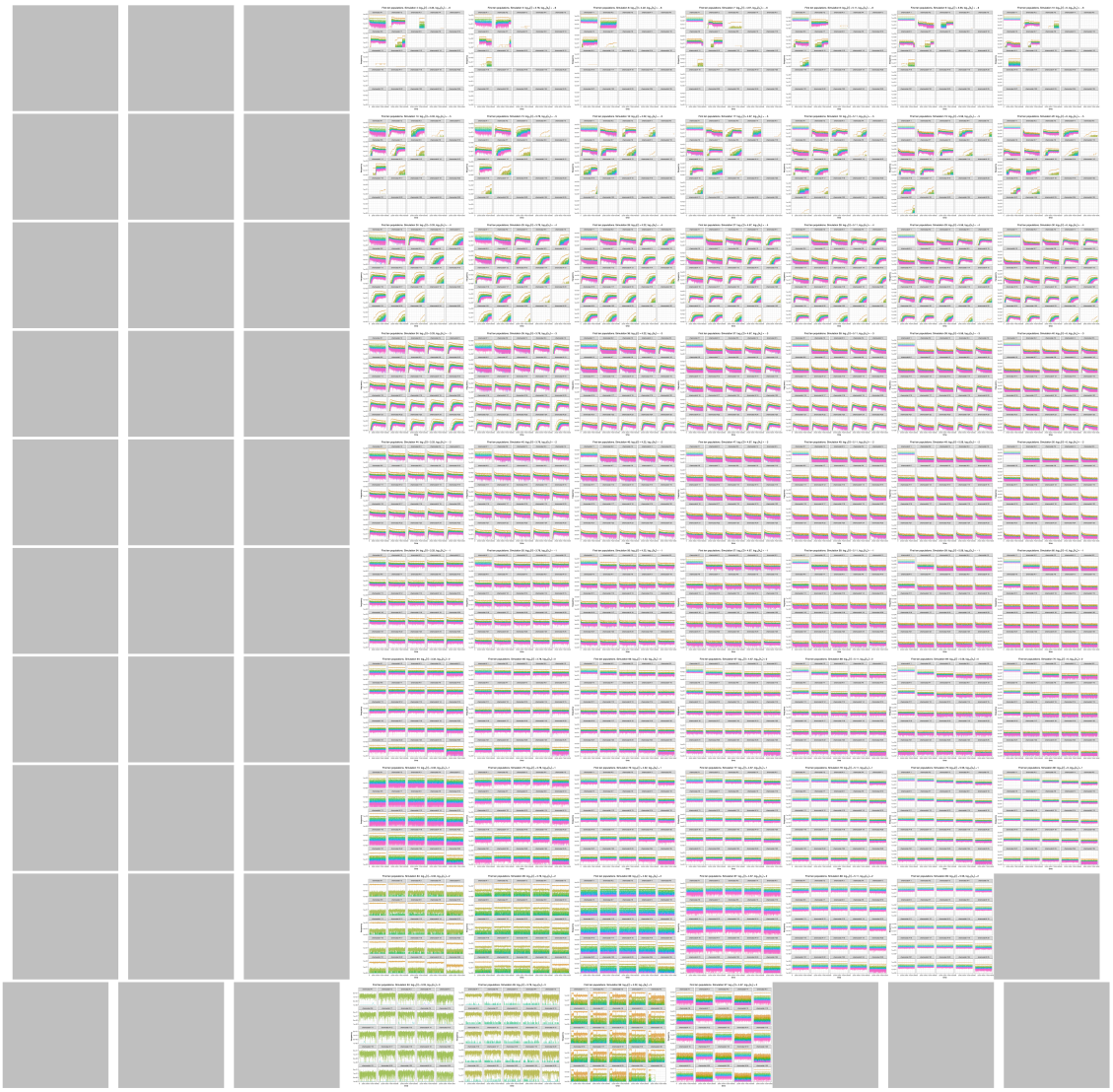

Figure 31: Same as prevous figure. Longer integration time.

{fig:MS25b}

Integration over 10^5 time steps
low resolution shown, original figure too heavy

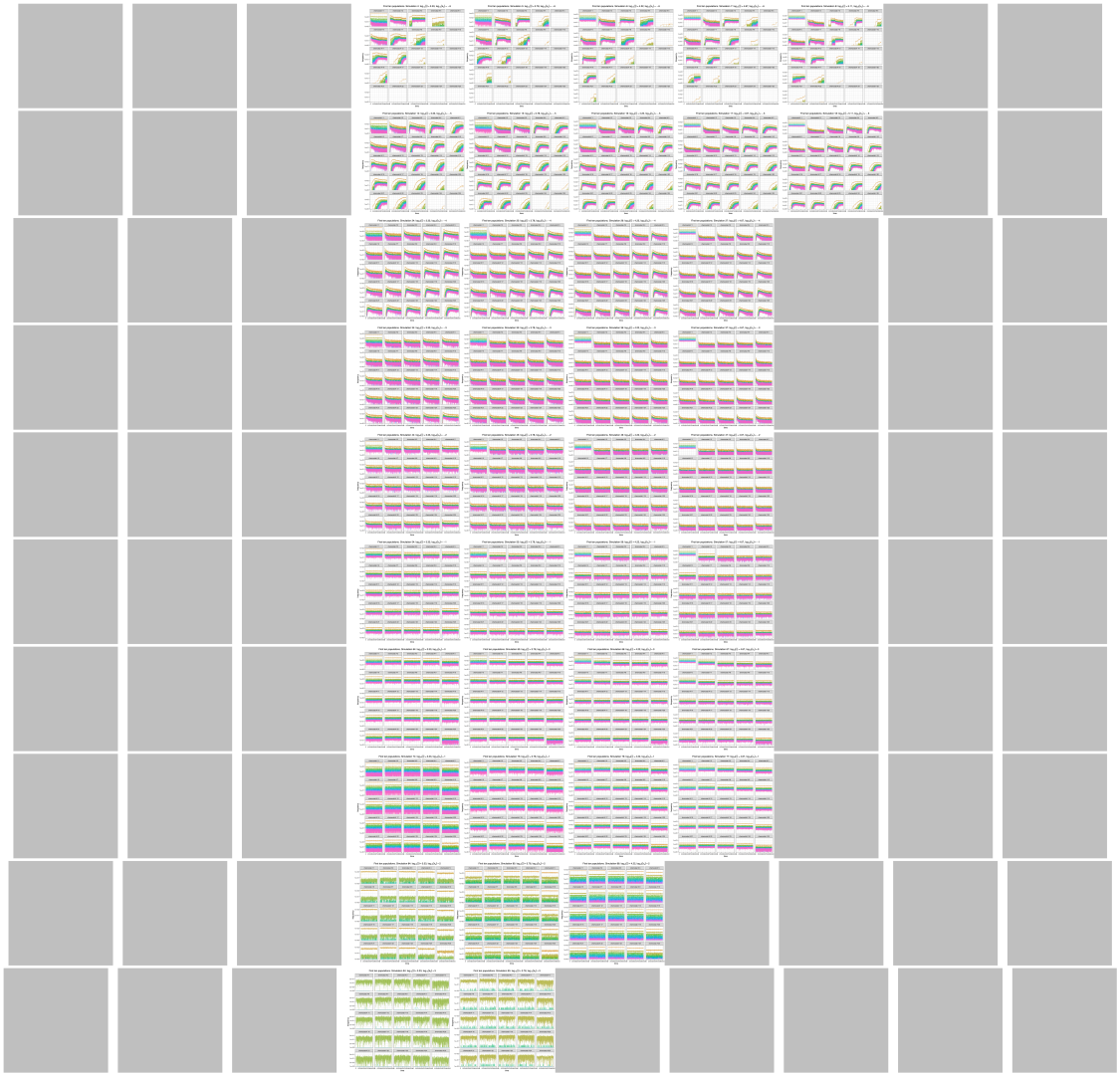

Figure 32: Same as two previous figures. Longer integration time.

{fig:MS25c}

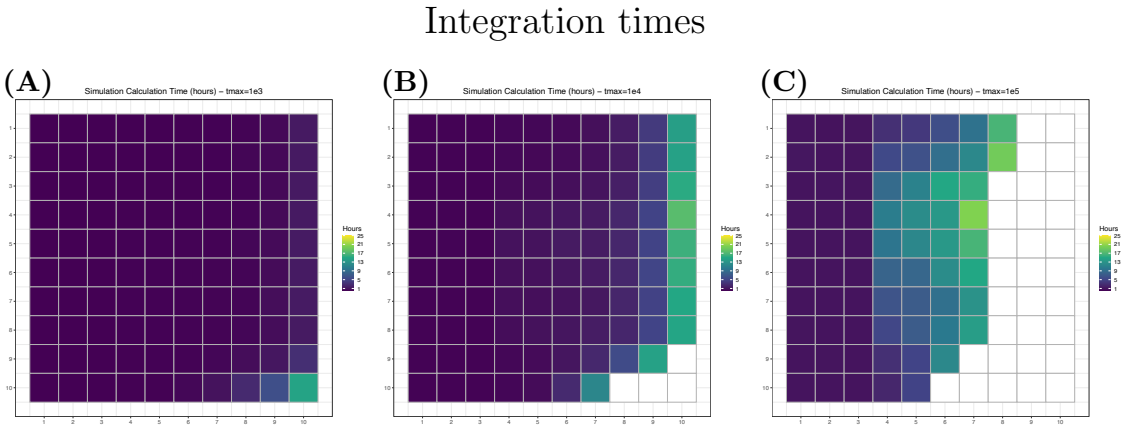

Figure 33: Integration time for the simulations in this dataset. White cases indicate simulations still running after 24 hours.

{fig:MS25d}

5.4 MS26: sweep over k_d with multiple inflows (2's only)

We now want to re-plot the population means vs k_d figure, this time showing only 2's and for a few different values of the inflow. From what we saw in the previous subsection, interesting ranges lie around $\log I = 3...6$ and $\log k_d = -3...1$. We'll integrate over 10^4 iterations which is sufficient to reach steady state in this regime. All other parameters stay the same: outflow equal to diffusion, forward rate 10^{-3}, lattice with $N = 25$ and so on.

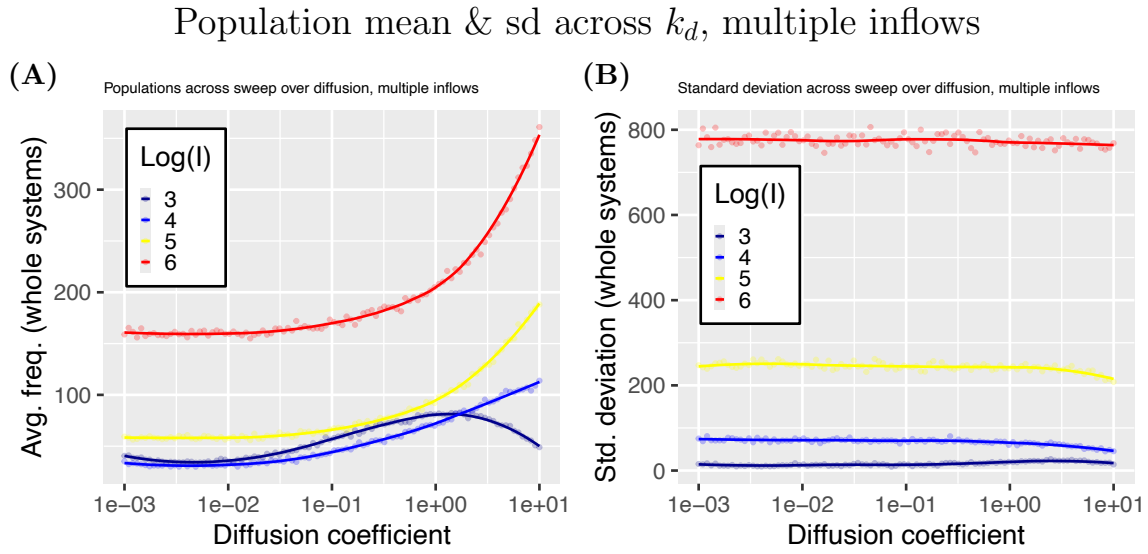

Figure 34: Population mean calculated across all reactors for a given simulation, and its standard deviation.

{fig:MS25d}

6 Datasets

6.1 Motivation

From now on, instead of creating a new "milestone" for every "experiment" we're doing I'll keep two separate (but hyperlinked) sections: *datasets* and *analysis*. The reason for this is that we're often going back and using previous batches of simulations to create new figures, which makes it rather confusing. Things will be kept much more clean by creating a new dataset each time I calculate a new batch of sims, and a new analysis each time we want to plot more figures. Obviously I'll clearly indicate for each analysis which datasets it sources the data from.

6.2 D01: sweep over k_d and I

In this dataset we do a sweep between $\log k_d = -1...3$ for 4 values of the inflow between $\log I = 3...4$. {subsec:D01}

Listing 1: content of `params.jl`

```
nrepeat = 1

inflow_rates = exp10.(LinRange(3,4,4))
diffusion_rates = exp10.(LinRange(-1,3,100))

params_template = Dict(
    # simulation parameters
    :save_interval      => 1e2,
    :method             => "tau-leaping",
    :dt                 => 1e-3,
    :random_seed        => "random",

    # physical parameters
    :total_time         => 1e4,
    :initial_mass       => 0,

    # topological parameters
    :graph_type         => "lattice-2way",
    :randomize_edges    => false,
    :N_reactors         => 25,

    # reaction rates
    :inflow_mols        => inflow_rates,
    :forward_rate       => 1e-3,
    :diffusion_rate     => diffusion_rates,
    :outflow_rate       => "equal-to-diffusion-rate",
)
```

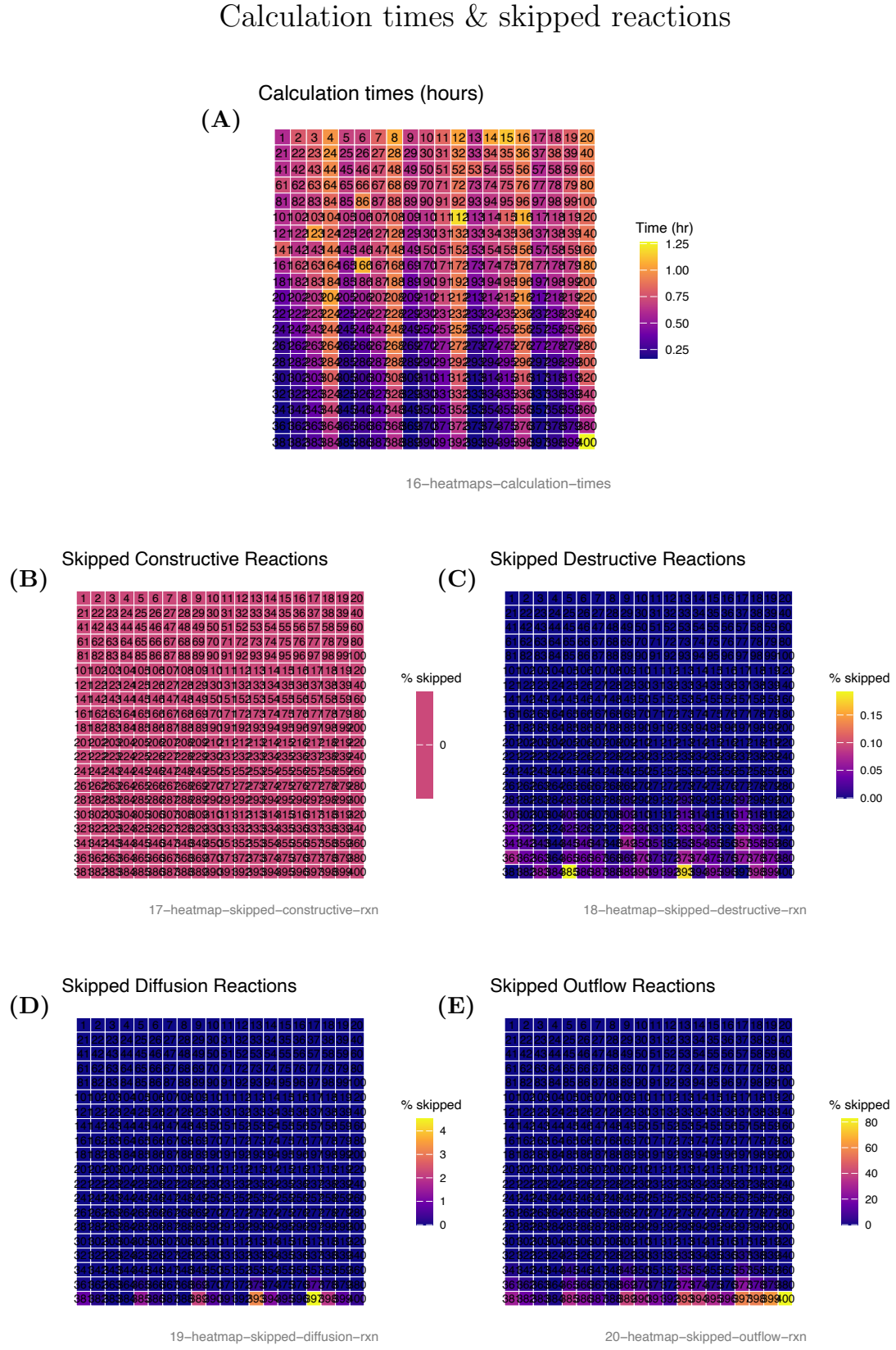

Figure 35: Diagnostic plots for D01.

{fig:D01}

6.3 D02: sweep over $\log k_d = -2...2$ with $\log I = 4$

In this dataset we do a sweep between $\log k_d = -2...2$ with the inflow $\log I = 4$. {subsec:D02}

Listing 2: content of `params.jl`

```
nrepeat = 1

params_template = Dict(
    # simulation parameters
    :save_interval      => 100,
    :method             => "tau-leaping",
    :dt                 => 1e-3,
    :random_seed        => "random",

    # physical parameters
    :total_time         => 1e4,
    :initial_mass       => 0,

    # topological parameters
    :graph_type         => "lattice-2way",
    :randomize_edges    => false,
    :N_reactors         => 25,

    # reaction rates
    :inflow_mols        => 1e4,
    :forward_rate       => 1e-3,
    :diffusion_rate     => exp10.(LinRange(-2,2,100)),
    :outflow_rate       => "equal-to-diffusion-rate",
)
```

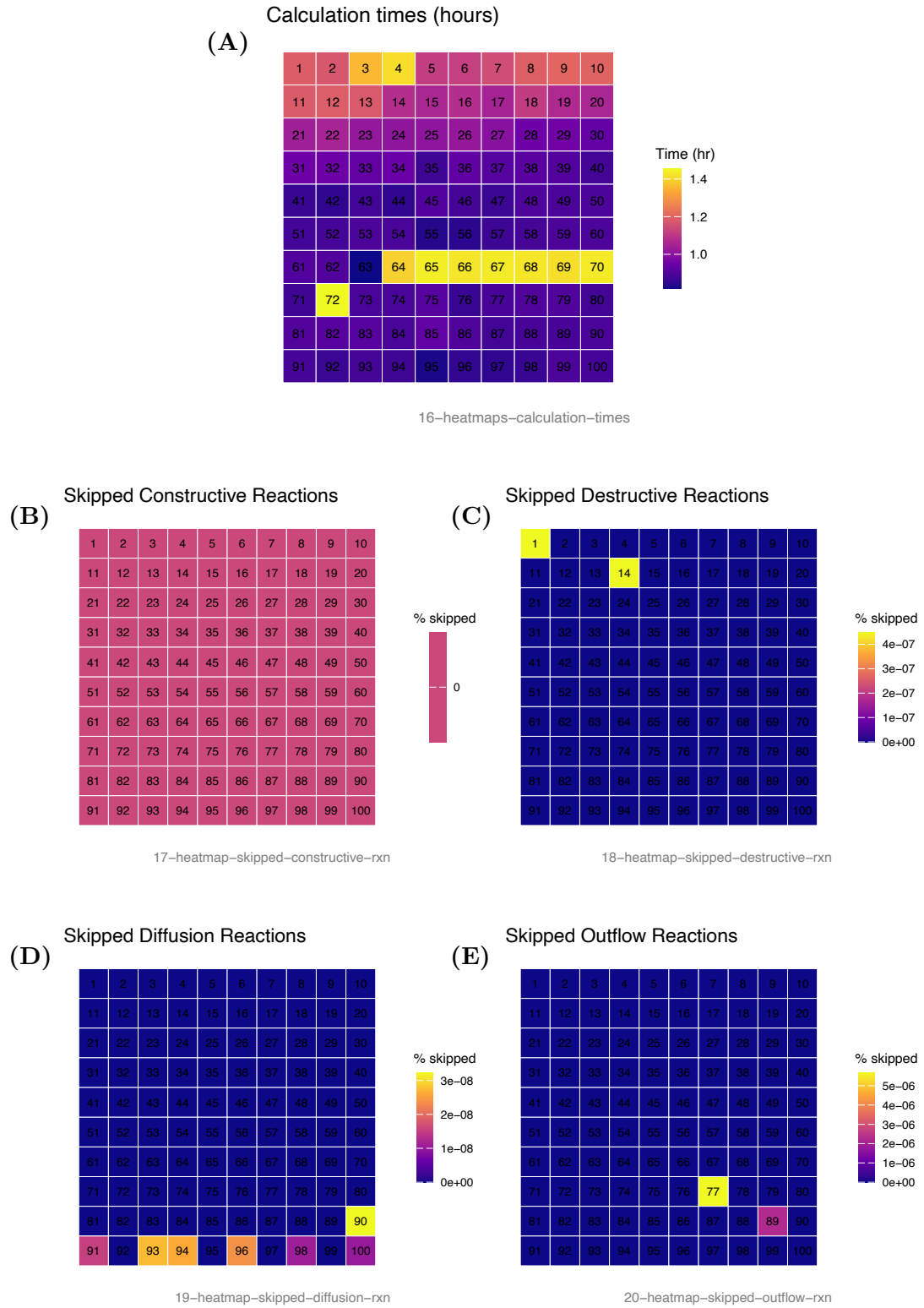

Figure 36: Diagnostic plots for D02.

{fig:D02}

6.4 D03: idem, randomized topology

Same thing as D02, but with randomized edges. {subsec:D03}

Listing 3: content of `params.jl`

```
nrepeat = 1

params_template = Dict(
    # simulation parameters
    :save_interval      => 100,
    :method             => "tau-leaping",
    :dt                 => 1e-3,
    :random_seed        => "random",

    # physical parameters
    :total_time         => 1e4,
    :initial_mass       => 0,

    # topological parameters
    :graph_type         => "lattice-2way",
    :randomize_edges    => true,
    :N_reactors         => 25,

    # reaction rates
    :inflow_mols        => 1e4,
    :forward_rate       => 1e-3,
    :diffusion_rate     => exp10.(LinRange(-2,2,100)),
    :outflow_rate       => "equal-to-diffusion-rate",
)
```

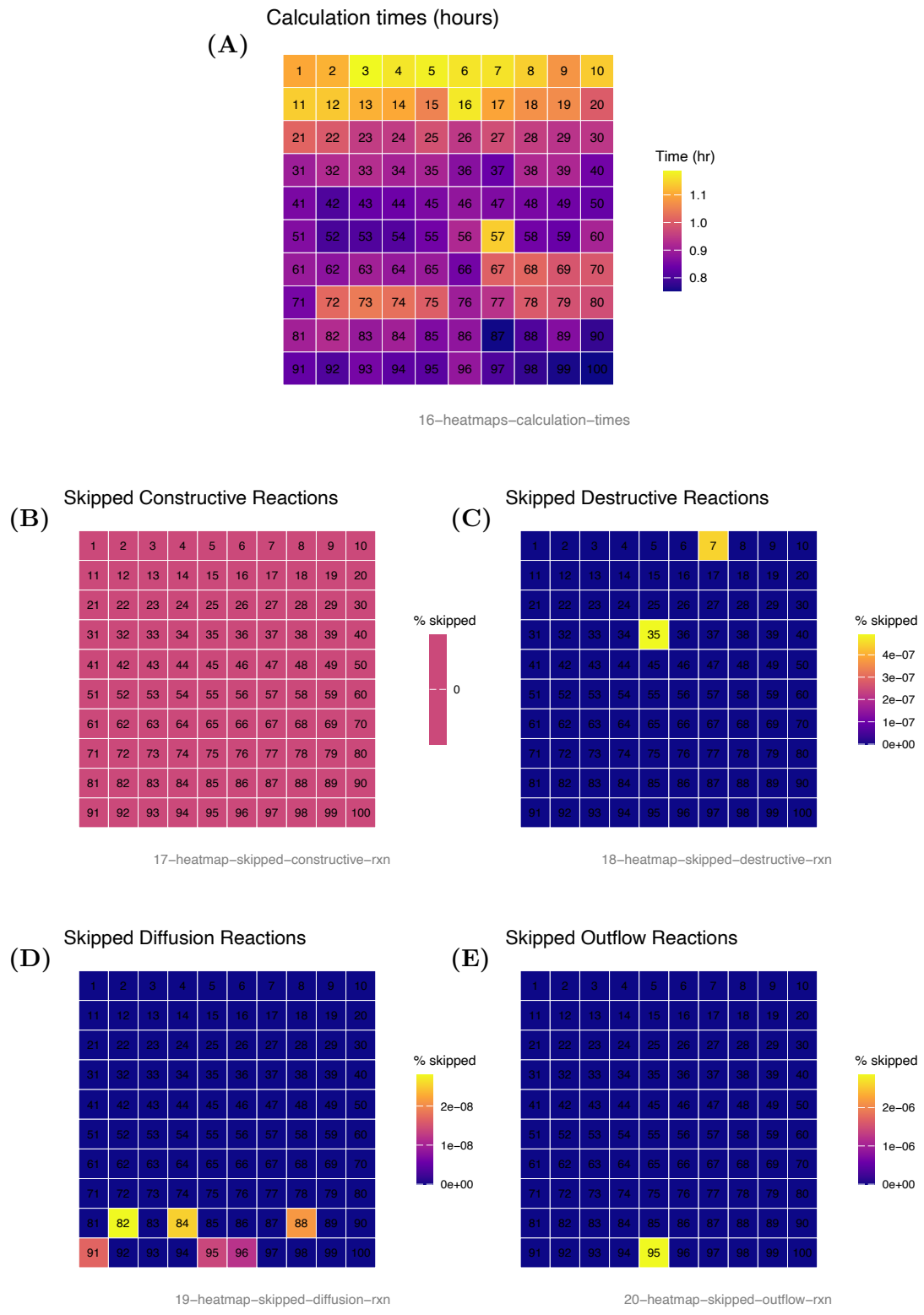

Figure 37: Diagnostic plots for D03.

{fig:D02}

7 Analysis

7.1 A01: plotting populations of 2's for a sweep over k_d and I (D01)

This analysis is based on dataset D01, which is a sweep over k_d and I. We're plotting the populations of 2's only (averaged over whole systems) over a range of diffusion parameters, for a few different values of the inflow.

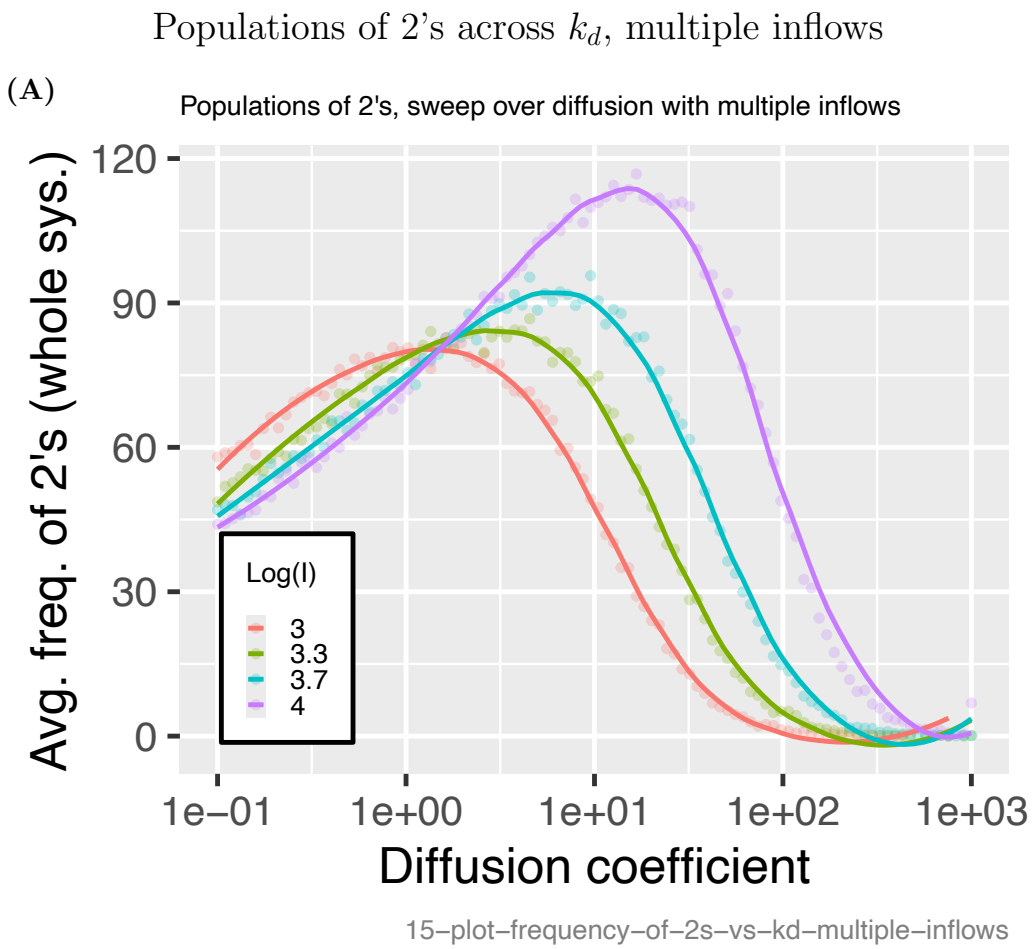

Figure 38: Populations of 2's over a range of diffusion coefficients, for a few values of the inflow.

{fig:MS25d}

7.2 A02: "wrap-up" figures for D02

We're focusing on $\log k_d = -2...2$ with a fixed inflow $\log I = 4$ in dataset D02. The goal will be to calculate two identical datasets, one with a lattice topology and the other with randomized edges. We are repeating the same figures we've plotted previously in MS15: time series for limit cases, power law fits over inflow/outflow PDFs, exponent of the fits as a function of k_d, average integer value and total mass. {subsec:A02}

One big difference here is that I have re-written the entire code for fitting the PDFs (panel E below): I have used the first 1000 integers, binned over 20 bins logarithmically spaced. Panel F shows the exponents of these fits for the inflow/outflow and the delta between them.

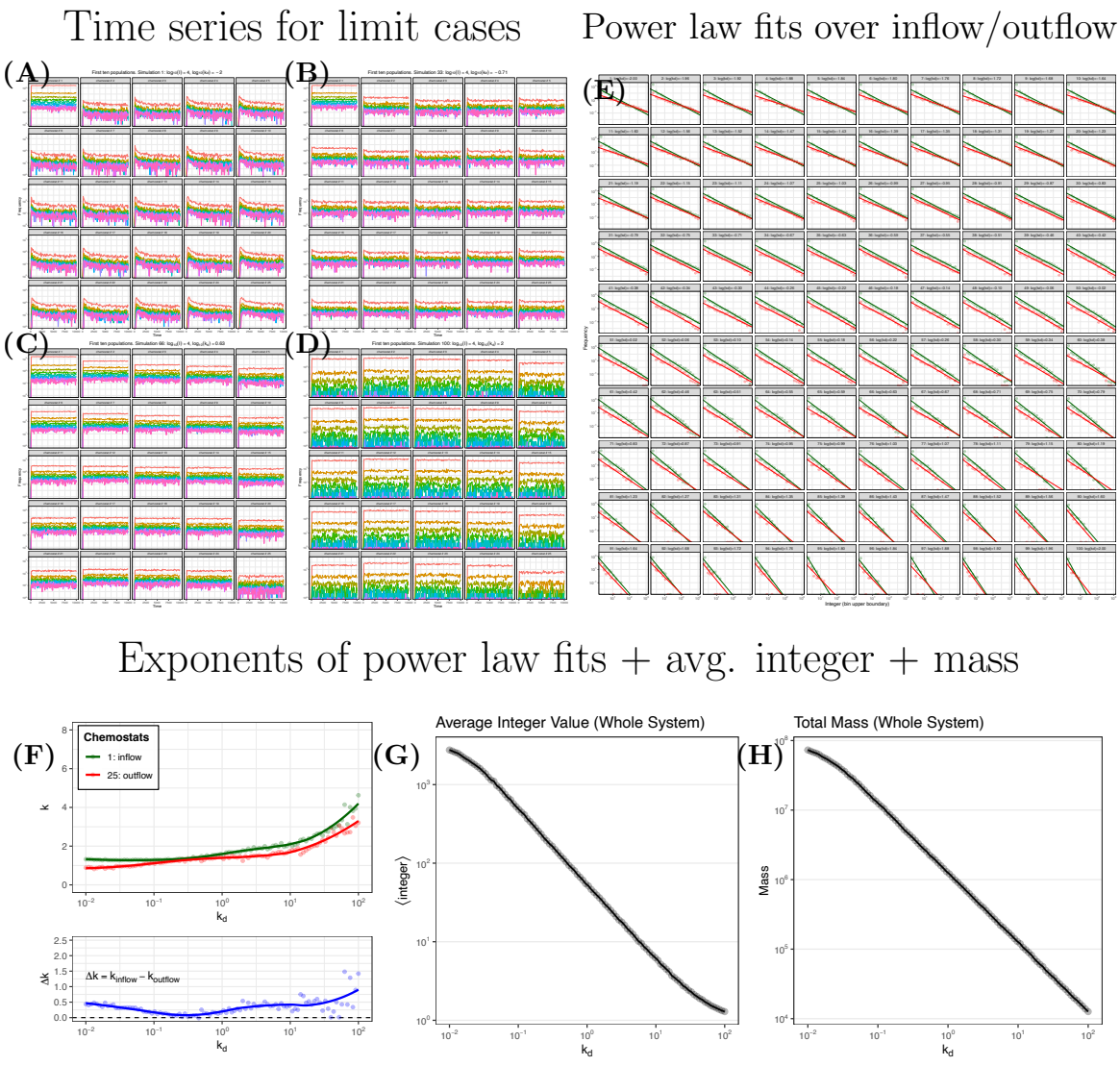

Figure 39: Key outputs: limit cases, power law fits and exponents, average integer and total mass.

{fig:wrap-up-15-18}

7.3 A03: "wrap-up" figures for D03

{subsec:A03}

Same thing as A02, but using a topology with randomized edges. Examining the time series for limit cases we can see that the reactors appear to be randomized (there's no general trend in species populations moving rightward/downward, like in the figures that used the lattice topology). Besides that, there isn't *much* difference—more fluctuations in average integers and mass but that's all in terms of *qualitative* differences.

However, if we examine closely panel F, we can see that even though the general trend is similar to that of the corresponding figure in A02, *the curve ends up at higher values of k on the right part of the figure*. In other words, randomizing the topology leads to higher exponents at high diffusion (where the topology matters more since the system is completely occupied), which means that the PDFs have a steeper curve, i.e. complex molecules are less present.

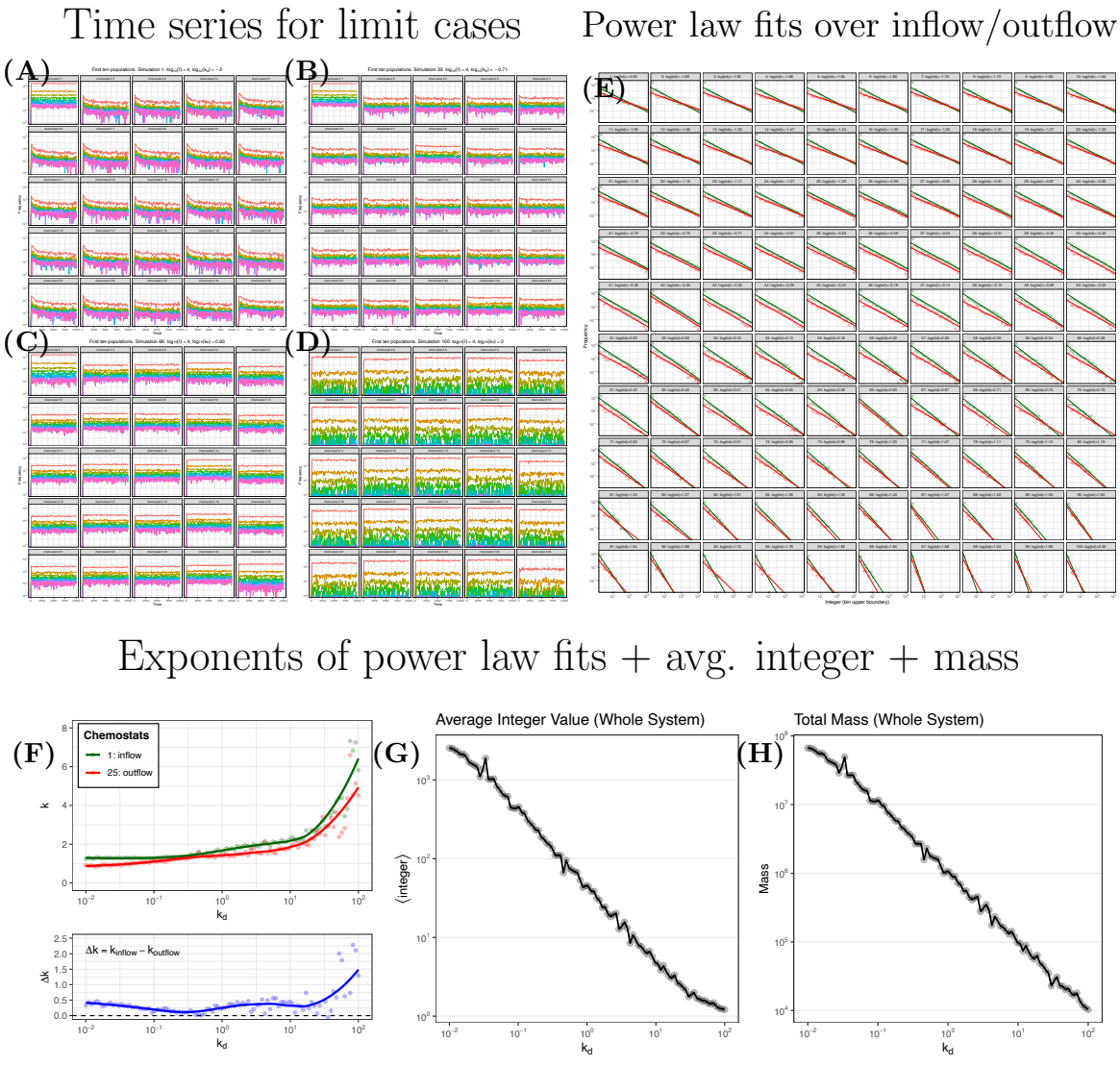

Figure 40: Key outputs: limit cases, power law fits and exponents, average integer and total mass.

{fig:wrap-up-15-18}

7.4 A04: punchline! Comparing lattice vs randomized PDF exponents.

In this analysis we're simply combining the outflow curves of panels F for A02 and A03 on the same figure. I have cropped the x-axis a bit and discarded points for $x < 10^0$ as well as adjusted the fit. I have also identified the regimes we're in (heterogeneous/well-mixed). This clearly shows that at high diffusion, where the topology matters more, randomizing the topology leads to *decreased* populations of complex molecules.

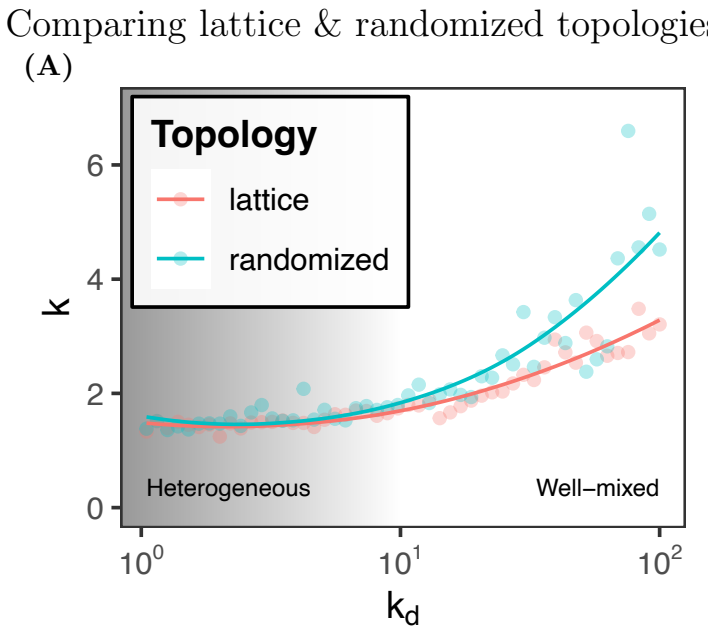

Figure 41: Exponents of the power law fits for outflow reactors: lattice vs randomized topologies.

{fig:MS25d}

References

Sharma A., Czegel D., Lachmann M., Kempes C. P., Walker S. I., Cronin L., 2023, Nature, pp 1–8