Mathematical models of diffusion-constrained polymerase chain reactions: basis of high-throughput nucleic acid assays and simple self-organizing systems

SUPPLEMENTAL INFORMATION

John Aach and George M. Church*

Department of Genetics and Lipper Center for Computational Genomics
New Research Building, Room 238
Harvard Medical School
77 Avenue Louis Pasteur, Boston, MA 02115, USA
Email: http://arep.med.harvard.edu/gmc/email.html
Phone: (617) 432-7562
Fax: (617) 432-6513

* to whom correspondence should be addressed

April 15, 2004

TABLE OF CONTENTS

1 Introduction

This document provides supplemental information for our main article “Mathematical models of diffusion-constrained polymerase chain reactions: basis of high-throughput nucleic acid assays and simple self-organizing systems.” Supplemental information is divided into two main sections: supplemental results not published in the main article, and additional details on algorithms. Supplemental results are provided in section 2, and algorithm details are provided in section 3.

We adopt the terminology and notation used in the main article. In particular, there are two classes of model: single polony growth models (SPGMs) and two polony interaction models (TPIMs). For each of these, there are three individual models that simulate polony growth and interaction in one, two, or three dimensions (symbolized by 1D, 2D, and 3D, respectively). The molecular species and algorithm parameters considered by the models are described in Tables 1 and 2 of the main article.

Equations, tables, and figures in this document are prefixed by “E”, “T”, and “F”, respectively, followed by the section number, and then suffixed with –1, –2, etc. For bookkeeping purposes, many figures and some text passages mention the names of files which are the sources of the data used to create the figures or support the discussion in the text. Although these may not be intrinsically meaningful to the reader, they will assist us in answering any questions which readers may pose concerning these figures.

2 Supplemental results

2.1 SPGM yield analysis

As noted in the main article, we have no mathematical proof that polony yield asymptotically develops power law behavior with exponent 1, 2, and 3 for the 1D, 2D, and 3D models. However, we note that the 2D and 3D cases can be reduced to the 1D case: This is because 2D and 3D diffusion equations in polar coordinates $\partial C/\partial t = \partial^2 C/\partial r^2 + (a/r)\cdot\partial C/\partial r$ reduce to a 1D Cartesian diffusion equation when r is sufficiently large because the $(a/r)\cdot\partial C/\partial r$ term drops out. Moreover, if it is accepted on the basis of Figure 4b in the main article that 1D polonies tend towards linear growth, the fact that T remains unchanged everywhere except at the edge of the polony implies that, in the limit, the polony enlarges by a constant increment per cycle of polony diameter. Therefore, since, in the limit, 2D and 3D polonies behave like 1D polonies, their diameters must also increase in constant increments per cycle, and thus must eventually exhibit quadratic and cubic growth, respectively.

Figure F2.1-1a expands on Figure 4a in the main article by showing $\log_{10}$ T yields over 100 cycles for the four SPGM simulations analyzed in Figure 4b; this includes standard parameter simulations (Table 2 of main article) for a 2D 10 μm thick gel, a 2D 1 μm thick gel, and a 1D model in addition to the 3D polony model shown in Figure 4a. Graphed with the $\log_{10}$ T yields are the compound exponential / power law regressions of these yields computed according to equation (1) of the main article. The regressions indicate that the dimensionality of the polony model affects the breakpoint between exponential and power law growth. Unsurprisingly, the transition occurs earlier in lower dimensions, doubtless due to the fewer degrees of freedom available for polony access to fueling species P and Q in lower dimensions. Specifically, the 1D simulation breaks into power law growth at cycle 7.8, the 2D 1 μm gel at cycle 15.0, and the 3D 10 μm gel at cycle 18.7, compared to the 3D breakpoint at cycle 23.3.

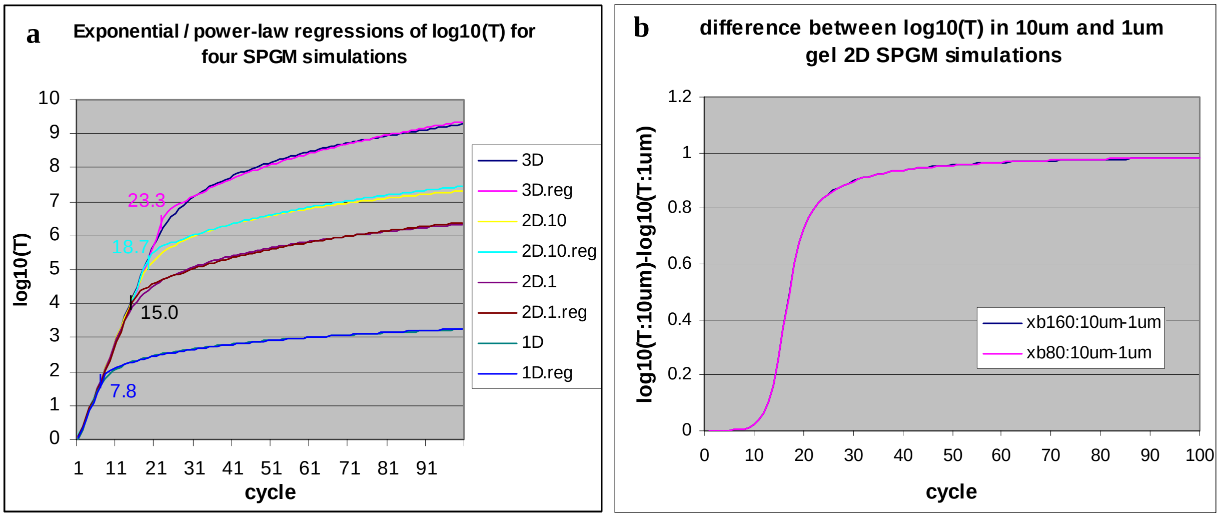


Figure F2.1-1: Additional SPGM yield analysis. (a) $\log_{10}$ T yields and compound exponential / power law regressions after denaturing cycles for four SPGM 100 cycle simulations with standard parameters (Table 2 of main article): 3D, 10μm thick 2D gel, 1μm thick 2D gel, and 1D, showing the breakpoints between exponential and power law growth as determined by the regressions. (b) Difference between $\log_{10}$ T yields for the 10μm and 1μm thick 2D gel SPGM simulations, showing that the 10μm simulation asymptotically approaches a 10-fold higher yield (1 $\log_{10}$ unit) compared to the 1μm gel. Additional discussion in text. Log_{10} differences for two pairs of simulations at two different algorithm spatial resolutions (see section 3.4) are seen to have identical results.

Reference file: pub.polony2D.dx_25.xb80.xb160.c100.2.outtotals.compare.xls

Formally, the concentrations generated by any 2D 10μm simulation should be calculable from the corresponding 2D 1μm concentrations simply by multiplying by 10; this follows from the SPGM equations (Figure 2a of main article) in conjunction with the 10-fold differences in rate constants, and ambient initial P and Q concentrations, and Q boundary conditions (i.e., iP and eQ, see Table 2 of main article) called for by the 10-fold difference in gel thickness. What prevents this from actually happening is that, unlike these other model parameters, the initial conditions at the polony seed location for the 10μm and 1μm simulations do not have the 10-fold difference required for complete correspondence: Rather, both simulations start with numerically identical concentrations of ST at the seed location that represent a single ST molecule, and likewise reduce P by exactly one molecule there. Complete correspondence resulting in the 10μm simulation having 10-fold concentrations relative to the 1μm simulation would require that the 10μm simulation start with 10 molecules of ST and a 10 molecule reduction of P. From this equivalent starting point, the 10μm and 1μm simulations undergo identical exponential growth, but, with 10-fold less P and Q available to support polony growth, the polony in the 1μm gel exhausts the supply P available in its interior earlier than the 10μm gel polony and breaks into power law growth ahead of it. Conversely, the 10μm polony needs a few more cycles to consume the extra P available to it to reach this point of polony maturity (see main article). These trends are clearly visible in Figure F2.1-1a, where it is also indicated that the difference in time needed to reach maturity is about 4 cycles. During this period, the growth of the 1μm polony slows from exponential to power law while the 10μm polony is still exponential, effectively allowing the 10μm polony to make up much of the difference between its actual 1 ST molecule starting point and the 10 ST molecule initial condition that would have supported complete correspondence with 1μm polony evolution. Consequently, at around cycle 19, when both polonies have reached maturity, the $\log_{10}$ T yield curves have separated to a distance of about 1 $\log_{10}$ unit, meaning that T yield in the 10μm polony is now nearly 10-fold more than that for the 1μm polony. At this point, both polonies now have expanding growth faces (see main article), and both are approaching power law growth with exponent 2, with the 10μm polony having a slightly higher exponent compared to the 1μm polony (Figure 4b in main article) and thus maintaining the 10μm polony at a slightly higher growth rate. As seen in Figure F3.2-1b, this slightly higher growth allows the 10μm polony to ever more closely achieve the state of complete correspondence with the 1μm polony, as the difference between the $\log_{10}$ T yields of the two polonies gets ever closer to 1.

2.2 SPGM polony morphology: effect of diffusion coefficients

One idea for increasing polony density in a gel is to decrease the amount of diffusion in the polony gel. This might also enhance polony exclusion (see main article) by making polony edges sharper, thereby decreasing the ability of neighboring polonies to invade each other at their edges. Both of these effects could also help alleviate an issue that arises when polonies are generated from sample DNA libraries containing DNA templates of different sizes: Since small templates generate much larger polonies than larger templates (Mitra and Church, 1999), polonies made from small template polonies in a library can take up an inordinate fraction of the gel area. One possible way to decrease diffusion through a gel is simply to increase the concentration of gel monomer in the solution that is polymerized to create the gel (Mitra and Church, 1999). Another way may be to use free primers with 'drag tags' that slow their movement through a gel. An advantage of the latter is that the gel density is not itself increased, leaving any substrates, products, and enzymes used in the polony PCR or subsequent assay reactions that do not have drag tags unchanged in their ability to diffuse and interact.

To gain preliminary insight into the idea of using drag tags, we ran a series of 4 3D SPGM simulations in which the diffusion coefficient of free primer D_Q was decreased 10-fold from its standard parameter value of 20μm^2/sec (Table 2 of main article). Results are shown in Figure F2.2-1. Decreases in D_Q had negligible effects on polony size, yield and morphology as gauged by tethered strand T. As seen in F2.2-1a, the T distributions at cycle 40 were nearly identical for all D_Q. Figure F2.2-1b shows that these decreases did affect polony yield and morphology as gauged by free strand S, however. Lowering D_Q makes it harder for Q to diffuse in from the polony edge to replenish Q that has been converted to S, making the overall S yield lower, but enough Q gets in to make enough S to convert P to T at normal rates. This is true even at the lowest D_Q used in the simulation series (.02μm^2/sec). Here the S distribution is not very different from the T distribution (see Figure F2.2-1d) and the large abundance of S over T noted in the main article vanishes completely.

However, the effects of interest—decreasing polony size and sharpening of the polony edge—are unlikely to be direct consequences of decreases in D_Q. Rather, any drag tag applied to Q will be incorporated into Q's extension product S and lead to decreases in the diffusion coefficient of free strand D_S. The simulations summarized in Figure F2.2-1 do not account for any such decrease. The extent of the decrease in D_S that would be caused by a drag-tag induced drop in DQ is presently unknown. Therefore, to explore this effect, we ran two 3D SPGM simulations in which D_Q remained at the lowest value in the prior series (.02μm^2/sec), but where, in addition, D_S was dropped 10-fold to .0035μm^2/sec and again to .00035μm^2/sec, compared to its standard value

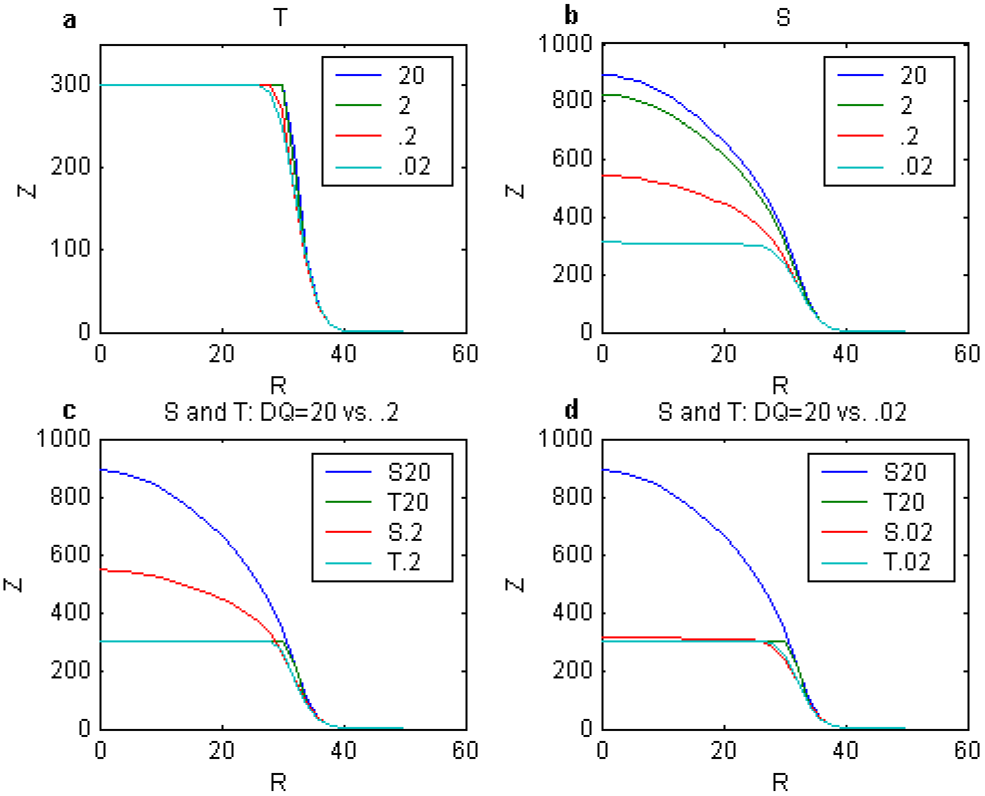


Figure F2.2-1: Effect on polony morphology of decreases in free primer diffusion coefficient D_Q. A series of four 3D SPGM simulations were run for 100 cycles differing only in D_Q, which was set at 20, 2, .2, and .02 μm^2/sec; standard values (Table 2 of main article) were used for all other parameters. Shown are R-Z plots of S and T after denaturing in cycle 40 from all simulations: (a) T concentration profiles are nearly identical for all simulations. (b) S concentrations are more depressed in the polony interior as D_Q diminishes. (c) and (d): Superimposition of S and T from D_Q = 20 with S and T from D_Q = .2 (c) and .02 (d).

Reference file: polony3D.DS_035.DQseries.outdata.xls

.035μm^2/sec. With the D_Q = .02μm^2/sec simulation from the prior series, in which D_S had its standard value, we could now compare three simulations with varying D_S at this low value of D_Q. The results are in Figure F2.2-2. Unlike the prior simulation series in F2.2-1, in which D_Q varied while D_S remained the same, polony size and yield now does vary considerably across changing D_S values, as polonies with a lower D_S grow more slowly. To aid comparison, we therefore synchronized this series by finding, for each cycle in a given simulation, the cycle in the other two simulations that had the same T yield after the PCR denaturing phase. Figure F2.2-2 shows the S and T distributions for one set of cycles which are matched by T yield in which the D_S=.035 (standard value) simulation data is taken from cycle 24, the D_S=.0035μm^2/sec simulation data from cycle 40, and the D_S=.00035μm^2/sec simulation data from cycle 99. This means that the D_S=.0035μm^2/sec simulation is being examined after the standard number of cycles (40) and the D_S=.035μm^2/sec simulation is being examined shortly after polony maturity. Figure F2.2-2a shows that, for polonies of equivalent yield, lowering D_S does indeed generate a polony with a sharper edge. Figure F2.2-2b shows the effect of the lower D_S on the S concentration distribution. Here, unlike the D_Q series above (Figure F2.2-1), lowering D_S appears to lead to increased S concentrations in the polony interior, but this is merely a consequence of looking at S at later cycles for series with lower D_S so that S has had more time to build up.

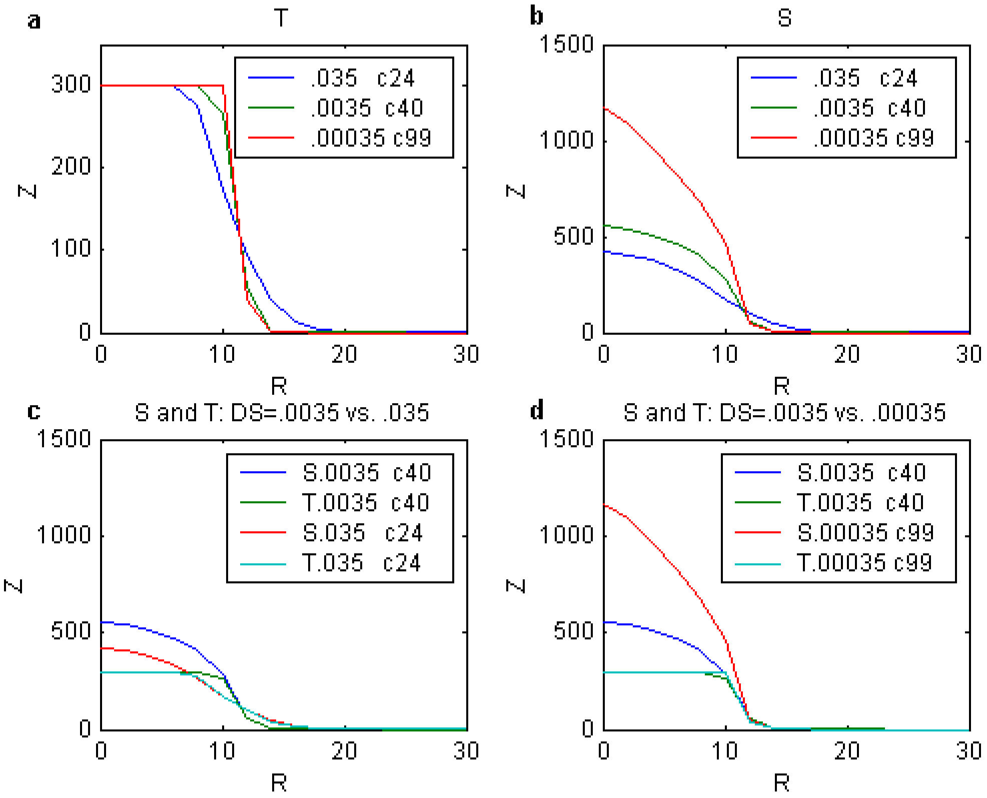


Figure F2.2-2: Effect on polony morphology of decreases in free strand diffusion coefficient D_S. A series of three 3D SPGM simulations were run for 100 cycles differing only in D_S, which was set at .035, .0035, and .00035 µm^2/sec; D_Q was set to .02 µm^2/sec and standard values (Table 2 of main article) were used for all other parameters. Shown are R-Z plots of S and T after denaturing cycles which showed closely matching T yields: cycle 24 for D_S=.035µm^2/sec, cycle 40 for D_S=.035µm^2/sec, and cycle 99 for D_S=.00035µm^2/sec: (a) T concentration profiles. The polony edge is sharper when D_S is lower. (b) S concentrations are larger in the polony interior as D_S diminishes because S is examined at later cycles for lower D_S. (c) and (d): Superimposition of S and T from D_S = .0035 with S and T from D_S = .035 (c) and D_S = .00035 (d).

Reference files: pub_polony3D_DQ_.02_DSseries_outdata.xls and pub_polony3D_DQ_.02_DSseries_outtotals.xls

2.3 1D TPIM: Analysis of 'tunneling'

The phenomenon designated 'tunneling' in the main article refers to a situation seen in 1D TPIM simulations where the concentration of an invading tethered strand V appears to drop in the midst of the invaded T polony but then to rise again on the far side of the T polony. An example was given in Figure 7d of the main article for polonies whose seeds were at +/- 4µm from the origin. There we mentioned that tunneling appears to reflect transient conditions in effect at the advancing T polony edge during the course of polony development that are preserved in the invading V spatial concentration profile. Evidence of this assertion is offered in Figure F2.3-1. It is seen that the spatial concentration profiles of T and invading strand V at cycle 40 reflect the

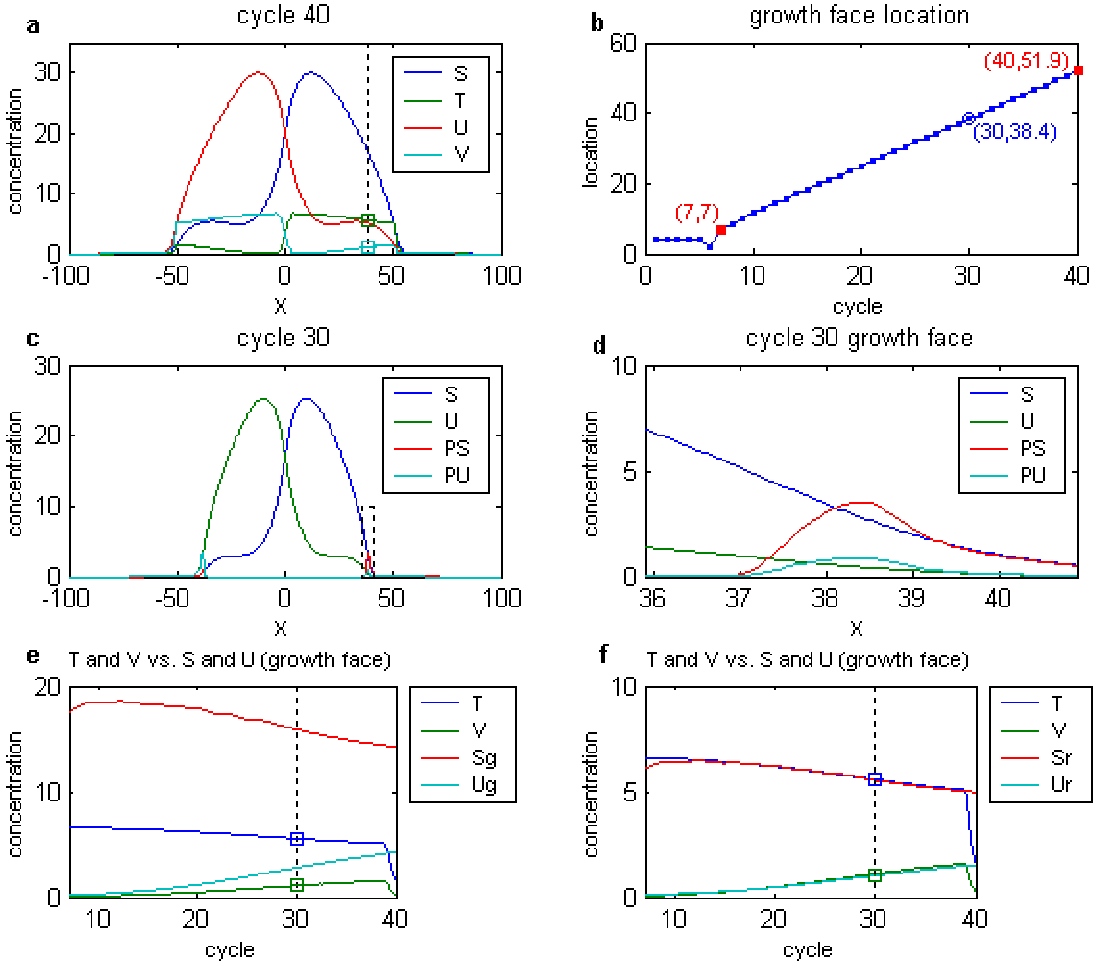


Figure 2.3-1: Analysis of 1D TPIM tunneling in relation to polony growth. (a) Example of tunneling. S, T, U, and V are shown after denaturing cycle 40 for a 1D TPIM simulation of polonies whose seeds were at +/- 4µm from the origin (same situation as Figure 7d of main article). Invading strand V is rising in concentration on the far side of the T polony at the dashed line, which represents where the T polony growth face was in cycle 30. Square markers: T and V concentrations at this location. (b) Graph depicting the location of the T polony growth face for each cycle, identified as the location at which the sum of PS and PU concentrations is maximal after every annealing phase. Red markers: beginning and end of apparent linear range. Blue marker: location of polony growth face in cycle 30 corresponding to dashed line in (a). (c) Depiction of cycle 30 free strand S and U concentration profiles (after denaturing) with respect to PS and PU (growth face) profiles (after annealing). For every cycle, the T growth face is taken to be contained in the region +/- 2.5µm from the location of maximum PS + PU concentration, indicated for this cycle 30 graph by dashed box. (d) Blow-up of cycle 30 T growth face region (dashed box) in (c). (e) Relation between amount of S and U in the T growth face to final cycle 40 spatial concentration profiles of T and V. For every cycle, the amount of S and U in the T growth face region is computed, yielding Sg and Ug. Within the linear region of (b), cycle is linearly related to a growth face location. Using this relation, the cycle 40 T and V spatial concentration profiles (seen in (a)) can be expressed as a function of cycle and put on the common abscissa 'cycle' with Sg and Ug. Dashed line indicates cycle 30 and corresponds to the dashed line in (a), and the square markers correspond to the marked values of T and V in (a). Sg and Ug at this line (see markers) are equal to the areas under the S and U curves depicted in (d). The Sg curve drops along with T, and the Ug curve rises with V. (f) The Sg and Ug curves of (e) are linearly rescaled to yield Sr = A·Sg + B and Ur = A·Ug + B, where A and B are found by least squares regression against the T and V values of (e). A very close match of Sr to T and Ur to V is observed. Regression solutions: A ≈ 0.345 and B ≈ 0.064.

Reference file: plny1De.DS_035.DQU20.c40.tba45.tbd30.dx_05.MPE3.4um.retest.hires.facedata.xls amount of free strand S and U that was in the advancing growth face of the T polony at each cycle. As described in the main article, the growth face comprises the small region at the edge of a mature polony at which new tethered strand is generated through hybridization during annealing of free strand with tethered primer. In TPIM simulations, the two hybrid species PS and PU must be considered. Figure F2.3-1 is consistent with the reasonable hypothesis that the rise of invading strand V on the far side of the T polony must be a result of a rising amount of U with respect to S in the growth face on the far side of the T polony over time, which allows an increasing amount of PU to form at the expense of PS as the polony grows. This rising amount of U is preserved in the V spatial concentration profile at later cycles and exhibits the phenomenon of tunneling.

What is not explained in this analysis is *why* U grows relative to S in the T growth face as the polony grows. One idea is that when there is substantially more T than V interior to the edge of the T polony growth face, this allows more growth face S to be 'wicked away' into unproductive hybrid ST than growth face U into unproductive UV. This may leave an increasing amount of U compared to T to be available to form productive hybrid PU vs. PS in the growth face, generating increasing concentrations of V compared to T in the growth face in subsequent replication phases. At this time, however, we have yet to confirm this hypothesis.

3 Algorithm details

The main work of the six polony models described in the article is to solve systems of diffusion and reaction-diffusion partial differential equations (PDEs). The purpose of this section is to provide additional detail about the algorithms beyond the brief descriptions given in the article, and also to characterize the accuracy of these algorithms. Accuracy was carefully examined because some of the results described in the article were counter-intuitive, e.g. 'tunneling' exhibited by 1D TPIM simulations. We wanted assurance that these results were not due to numerical artifacts generated by the algorithms or our implementations of them.

3.1 Implicit algorithms used by polony model packages

Our main reference for PDE solution algorithms was (Ames, 1992), supplemented by (Press, et al., 1996). Use was also made of (Crank, 1956) when PDEs involved polar coordinates. Below we summarize the algorithms used in each of the six models. Further details can be determined directly from the source code we have provided for each model on our supplemental web site http://arep.med.harvard.edu/polony_models/.

PDEs below are described for simple diffusion equations for a generic concentration variable C with a diffusion coefficient D and a boundary value b. The abbreviations dx, dz, dr, and dt refer to mesh sizes used to represent the continuous PDEs as finite difference equations. Different combinations of mesh sizes are used in different models as described below.

In the descriptions below, *Polony Seed Initial Conditions* refers to the concentration profile assigned to the initial single ST molecules used by the models; for TPIMs, these considerations also apply to initial single UV molecules. The designation 'standard initial conditions' means the following: Let dV = the volume element appropriate to the models coordinate system; only models that use exclusively Cartesian coordinates are considered here so that dV = dx for 1D Cartesian simulations and dx^2 for 2D Cartesian simulations. For $1/dV \leq iP$, then ST is set to 1/dV at the seed location and P is reduced from iP to iP-(1/dV) at that location. Under these conditions, ST is initialized to a pure discretized delta function at the seed location. Where $1/dV \geq iP$, a pure discretized delta function is not admissible and initialization is as described in Figure

F3.1-1. In short, under these conditions, the initial condition is as close to a pure delta function as possible given the constraint that initial ST concentration must be everywhere ≤ iP.

In every model, reaction-diffusion equations are solved by operator splitting as described in (Press, et al., 1996, chap. 19.3). Thus, for a reaction diffusion equation of the form $\partial C/\partial t = f(C,X\ldots)+D\nabla^2 C$, where f(C,X,…) is a function of C and other concentrations X at the given spatial coordinates, is solved by alternating between steps of time length dt of the simple diffusion equation $\partial C/\partial t = D\nabla^2 C$ and the equation $\partial C/\partial t = f(C,X\ldots)$ (which involves no spatial derivatives).

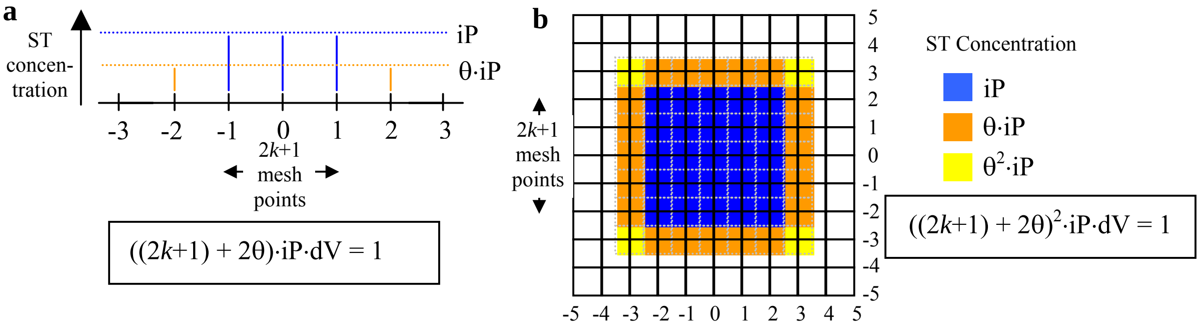


Figure F3.1-1: Standard initialization conditions for ST seed molecule when a pure delta function at the origin is insufficient. P is decreased by the concentration of ST at every mesh point. *k* is an integer and 0 ≤ θ < 1. (a) 1D polony model. (b) 2D polony model.

3.1.1 SPGM 1D (polony1D)

Solution space: 1D Cartesian variable x, discretized as x = i·dx where –xb ≤ i ≤ xb for integral values of i.

Algorithm: implicit equations with a variable parameter λ.. λ = 0.5 corresponds to Crank-Nicolson equations, λ = 1.0 corresponds to “O’Brien” equations described in (Ames, 1992, section 2.3, equation 2-33) and also to (Press, et al., 1996, chap. 19.2) (where, however, the name “O’Brien” is not cited). The polony1D default is λ = 1.0.

Equations:

$$\text{Parameters:}$$

$$r = dt/dx^2$$

$$\lambda = parameter\ (0 \le \lambda \le 1)$$

$$\text{(E3.1.1-1)}\quad \text{Equations:}$$

$$-rD\lambda \cdot C(i-1,t+1) + (1+2rD\lambda)C(i,t+1) - rD\lambda \cdot C(i+1,t+1) =$$

$$rD(1-\lambda)C(i-1,t) + (1-2rD(1-\lambda))C(i,t) + rD(1-\lambda)\cdot C(i+1,t)$$

$$C(-xb,t+1) = C(xb,t+1) = b$$

Polony seed initial conditions: standard initial conditions (see above).

3.1.2 SPGM 2D (polony2D)

Solution space: 2D Cartesian variables x and y, discretized as x = i·dx where –xb ≤ i ≤ xb for integral values of i, and as y = j·dx where –xb ≤ j ≤ xb for integral values of j. The variables x and y therefore have the same mesh size and bounds.

Algorithm: Alternating Directions Method for two dimensions as described by (Ames, 1992, equations 5-77).

Equations:

$$\text{Parameters:}$$

$$r = dt/dx^2$$

$$\text{Equations: Alternate between (for all j)}$$

$$(\text{E}3.1.2-1)\begin{cases} -\frac{rD}{2}C'(i-1,j,t+1)+(1+rD)C'(i,j,t+1)-\frac{rD}{2}C'(i+1,j,t+1)= \\ \quad\frac{rD}{2}C(i,j-1,t)+(1-rD)C(i,j,t)+\frac{rD}{2}C(i,j+1,t) \\ C'(-xb,j,t+1)=C'(xb,j,t+1)=b \end{cases}$$

$$\text{and (for all i)}$$

$$\begin{cases} -\frac{rD}{2}C(i,j-1,t+1)+(1+rD)C(i,j,t+1)-\frac{rD}{2}C(i,j+1,t+1)= \\ \quad\frac{rD}{2}C'(i-1,j,t)+(1-rD)C'(i,j,t)+\frac{rD}{2}C'(i+1,j,t) \\ C(i,-xb,t+1)=C(i,xb,t+1)=b \end{cases}$$

Polony seed initial conditions: standard initial conditions (see above).

3.1.3 SPGM 3D (polony3D)

Solution space: A single 3D radial coordinate r, discretized as r = i·dr where 0 ≤ i ≤ rb for integral values of i.

Algorithm: implicit equations with a variable parameter λ. (see section on SPGM 1D above), but based on the polar coordinate difference equations from (Crank, 1956, equations 10.41 and 10.42).

Equations:

$$\text{Parameters:}$$
$$a = dt/dr^2$$
$$\lambda = parameter\ (0 \le \lambda \le 1)$$
$$\text{Equations: i > 0:}$$

$$-aD\lambda\frac{i-1}{i}C(i-1,t+1)+(1+2aD\lambda)C(i,t+1)-aD\lambda\frac{i+1}{i}C(i+1,t+1)=$$

$$\text{(E3.1.3 − 1)}\qquad aD(1-\lambda)\frac{i-1}{i}C(i-1,t)+(1-2aD(1-\lambda))C(i,t)+aD(1-\lambda)\frac{i+1}{i}C(i+1,t)$$

$$\text{For i = 0:}$$
$$(1+6aD\lambda)C(0,t+1)-6aD\lambda\cdot C(1,t+1)=(1-6aD(1-\lambda))C(0,t)+6aD(1-\lambda)C(1,t)$$
$$\text{For i = rb:}$$
$$C(rb,t+1)=b$$

Polony seed initial conditions: The analog to standard initial conditions in this spherical geometry would be to use a delta function at the origin when 1/dV ≤ iP, and, otherwise, to spread out the initial ST molecule at concentration iP in the interior of sphere with a non-zero radius with any required remaining mass spread evenly over the spherical boundary. However, the above algorithm does not work at all with pure delta function initial conditions and so standard initial conditions cannot be used even when the 1/dV ≤ iP and no spreading out is necessary. Therefore standard initial conditions are abandoned entirely. Instead, the initial ST molecule is treated as if it were placed at the origin as a pure delta function and allowed to diffuse until its concentration is everywhere less than iP. P is then adjusted by everywhere subtracting the resulting ST concentration from P's initial iP concentration value. We call this process 'pre-diffusion'. Because the algorithm does not work with pure delta functions the result of ST's pre-diffusion is computed directly using the appropriate theoretical Gaussian distribution (Berg, 1993, equation 2.8). The pre-diffusion time is specified *via* a user parameter. If it is not sufficiently large the concentration of P may be negative at the origin, and in this case, the algorithm signals an error and terminates.

3.1.4 TPIM 1D (plny1De)

PDE algorithm is identical with that of the SPGM 1D algorithm in polony1D described above.

3.1.5 TPIM 2D (plny2De)

PDE algorithm is identical with that of the SPGM 2D algorithm in polony2D described above.

3.1.6 TPIM 3D (plny3De)

Solution space: 3D cylindrical coordinates as given by a 2D radial coordinate r and a 1D Cartesian coordinate z, where r = i·dx where 0 ≤ i ≤ rb for integral values of i, and z = j·dz where –zb ≤ j ≤ zb for integral values of j. The variables x and y therefore have the same mesh size and bounds.

Algorithm: Operator splitting as described in (Press, et al., 1996, chap. 19.3) applied to the 2D radial and Cartesian variable; i.e., each time step is processed as if diffusion took place first along the 2D radial variable and then took place for the Z Cartesian variable. Diffusion for the 2D radial variable uses a Crank-Nicolson version of (Crank, 1956, equations 10.36 and 10.37), while diffusion for the Z variable uses regular 1D Cartesian Crank-Nicolson equations (equivalent to the SPGM 1D equations E3.1.1-1 with $\lambda = 0.5$).

Equations: Only the equations for the 2D radial coordinate diffusion are given here. The equations for the Z coordinate are the same as E3.1.1-1 with $\lambda = 0.5$.

Parameters :

$$a = dt / dr^2$$

Equations : i > 0 :

$$-aD\frac{2i-1}{2i}C(i-1,t+1) + (1+aD)C(i,t+1) - aD\frac{2i+1}{2i}C(i+1,t+1) =$$

$$\text{(E3.1.6}-1) \qquad aD\frac{2i-1}{2i}C(i-1,t) + (1-aD)C(i,t) + aD\frac{2i+1}{2i}C(i+1,t)$$

For i = 0:

$$(1+2aD)C(0,t+1) - 2aD\cdot C(1,t+1) = (1-2aD)C(0,t) + 2aD\cdot C(1,t)$$

For i = rb:

$$C(rb,t+1) = b$$

Polony seed initial conditions: Unlike the 3D SPGM radial diffusion equations of polony3D, these finite difference equations do support delta functions at the radial origin. However, in cases where iP ≤ 1/dV, it is still necessary to spread the initial ST and UV density out beyond the seed locations themselves. The 3D cylindrical analog for standard initial conditions would create cylinders about the seed locations in which ST (UV) had constant concentration iP, and smaller concentrations along its borders, but these calculations are inconvenient and, given that the dr and dz coordinate mesh sizes are independently settable, they could lead to slightly different geometries from simulation to simulation. Therefore, as with 3D SPGM, standard initial conditions are again abandoned and a pre-diffusion approach is used. Therefore, in this model, since pure delta functions are supported, for each of ST and UV, a mass of 1 is placed at the seed location and the diffusion algorithm is run until the concentration at the seed location is ≤ iP. Here pre-diffusion is entirely automatic and does not require setting of a parameter as in the case of 3D SPGM. The resulting distributions are subtracted from the P distribution. The diffusion coefficient D_S is used for both the ST and UV, generating equivalent initial spatial distributions.

3.2 Algorithm accuracy

As noted above, we carefully examined algorithms for accuracy to provide assurance that results that appeared to be counter-intuitive were not due to artifacts in the algorithms.

The following means were generally used for testing algorithm accuracy: *Test A:* We attempted to verify that application of the algorithm to a pure diffusion problem with a known theoretical solution obtained results that are compatible with the solution. Generally this meant running the algorithm on an initial delta function, a problem whose theoretical solution is a Gaussian distribution. Since the first denaturing cycle of polony simulation results in the diffusion of a single S molecule, it is often possible to simply check the results of this cycle. Generally, we verified that graphs of computed and theoretical results superimpose well, but sometimes also examined numerical measures of accuracy (these are not straightforward because there is no simple numerical measure of overall accuracy). *Test B:* We could also verify that total S mass is conserved after denaturing cycles. *Test C:* We reran simulations with smaller spatial mesh sizes and verified that graphs of predicted molecule concentrations superimpose well and that changes in yields are small. This method was used to identify the mesh sizes used for most figures in the article.

The temporal mesh size dt was generally set to .01 sec. This was determined during very early testing of the first polony models by comparing the results of C code against a version coded in Mathematica 3.0 (Wolfram Software; Champaign, IL) (Figure F3.2-1). As Mathematica uses another PDE algorithm (the Method of Lines, according to Mathematica documentation), this comparison also helped establish that the implicit algorithm used in the C polony model was functioning correctly.

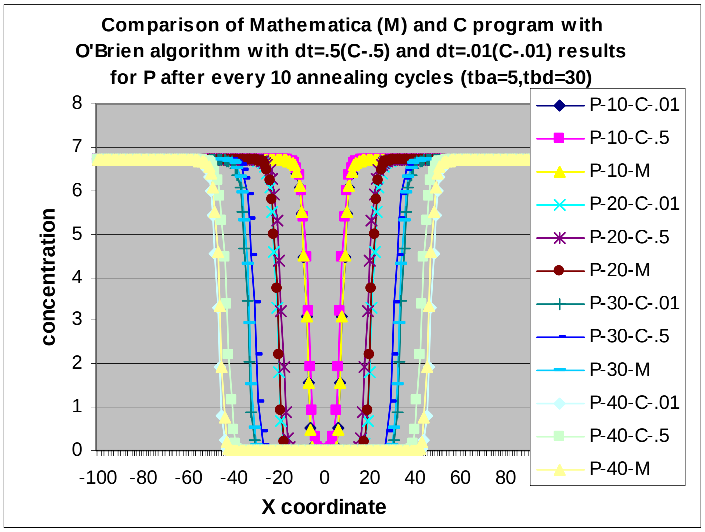


Figure F3.2-1: Comparison of results of early version of polony1D (1D SPGM) with Mathematica 3.0 implementation of model. Shown are the concentration profiles of P (tethered primer) after every 10 annealing cycles (where annealing time = 5 seconds). C program values with dt = .01 (C-.01) almost superimpose with the Mathematica results (M) whereas C program values with dt = .5 do not superimpose as well (C-.5).

polony1D.DS_035.DQ20.tba5.tbd30.c40.dt_01.compare.xls

An example of *Test A* is given in Figure F3.2-2. This figure overlays the initial S diffusion cycle as computed in the 3D TPIM (plny3De), the most demanding algorithm of all six, with the theoretically computed Gaussian.

When all is said and done, only *Test A* truly ensures that algorithm accuracy, as only this test shows that predicted results accord with theoretically correct results. *Test B* is at most a prerequisite for accuracy, and *Test C* is really a test of algorithm precision. As a test for accuracy, *Test A* is limited because it only demonstrates accuracy in one set of conditions that is only found at most once in the course of polony simulation. Therefore, in the end, *Test C* bears most of the burden for deciding the quality of a simulation, and precision is effectively a stand-in for accuracy. Several examples of *Test C* are given in section 3.4.

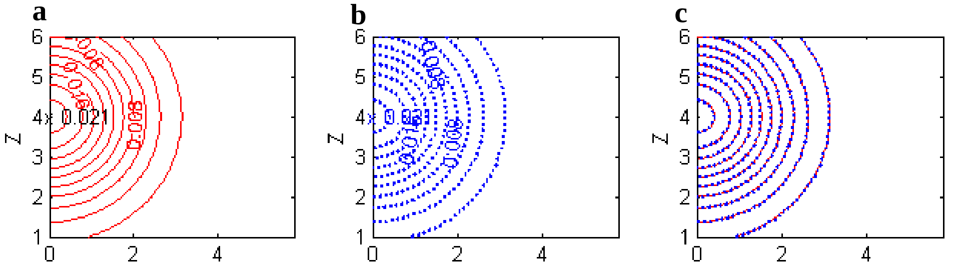


Figure F3.2-2: R-Z contour plots of S after the first denaturing cycle of a plny3De TPIM 3D simulation. The ST seed molecule was at +4 µm on the Z-axis. (a) Computed distribution. (b) Theoretical Gaussian distribution. (c) Superimposition of a and b.

plny3De.dr_05.rb200.dz_25.zb80.c40.z1.16.cyc1.erranalyze.fig

3.3 Dynamic rescaling

We developed dynamic rescaling logic to reduce the performance requirements needed to support accurate simulation of polony evolution in models requiring multiple spatial coordinates: 2D SPGM (polony2D) and TPIM (plny2De), and 3D TPIM (plny3De). Processing of multiple spatial coordinates is intrinsically expensive compared to single spatial coordinates, and is additionally penalized by generally requiring smaller spatial mesh sizes than single spatial coordinate models to achieve good accuracy.

The problem that motivated dynamic rescaling is this: Polonies begin with a single seed molecule in a small volume but grow to have dimensions perhaps 100x larger over the course of 40 cycles. To accurately capture the spatial characteristics of molecular species concentrations in early cycles requires small spatial mesh sizes dx, dz, or dr (depending on the model). However, because the polony is much larger at the end of 40 cycles, the models therefore require large solution space bounds xb, zb, or rb. This is intrinsically inefficient. While the polonies are small, it is only near the origin that concentrations differ meaningfully from initial conditions, so all the calculations expended on computing concentrations away from the origin generate information of no value. Later on, when the polonies are large, calculations away from the origin do generate important information, but the small mesh sizes maintain this information at a spatial resolution that is finer than needed.

These considerations apply to all polony models, whether they use single or multiple spatial variables. But single spatial variable models (1D SPGM (polony1D) and TPIM (plny1De) and 3D SPGM (polony3D)) perform well enough that the overhead of these wasted calculations is not onerous. However, for multiple spatial variable models, the overhead is prohibitive. Early results indicated that a 3D TPIM simulation using standard parameters with mesh sizes suitable for good accuracy could take over 10 days to run on a ~933 MHz machine. With dynamic rescaling, most of these simulations now take on the order of 4-8 hours.

The basic idea of dynamic rescaling is simple: Rather maintain a small spatial mesh size with a large solution space, use a solution space with a fixed, modest number of mesh points and increase the size of the mesh distances as the polony grows. For example, in a 2D SPGM simulation, rather than use a mesh size dx = .25µm but maintain an xb = 300 to ensure that the ~50µm radius polony generated at cycle 40 is adequately supported (including room to keep the polony well away from the boundary to ensure that PDE boundary conditions are met)—an arrangement that would require a solution space containing 601^2 coordinate pairs—start, instead, with dx=.25μm and maintain xb = 80 over the course of the simulation (for a space of 161^2 coordinate pairs) by increasing the size of dx as the polony grows to achieve the same result with only 7% of the work $((161/601)^2)$.

Dynamic rescaling is controlled through four parameters:

threshold: a magnitude that is used to identify the edge of the polony. It defines the concentration of S (or, in TPIMs, S or U) above which a point in solution space is considered to be within the polony.

trigger: a fraction of a coordinate boundary that is used to direct when rescaling is to take place after a denaturing cycle. Continuing the 2D SPGM example above, where xb = 80, after each denaturing cycle, the algorithm looks for the largest absolute value of an x or y coordinate whose S value exceeds *threshold*, and computes the fraction between this value and 80. If this fraction exceeds *trigger*, a rescaling operation is initiated.

abort: When, after a certain denaturing cycle, fractional coordinate bounds are found to exceed *trigger*, it means that the polony has grown large enough to cover a substantial portion of solution space. A rescaling operation is then needed to give the polony more growing room so as to keep it away from the solution space boundary. However, it is also possible that when the polony first exceeds *trigger* it may already have come so close to the boundary to violate the safe distance needed to ensure diffusion equation boundary conditions. The *abort* value is a threshold fractional coordinate bound beyond which the polony is considered to have grown too close to the boundary and the simulation should be considered invalid: If the fractional bound exceeds *abort*, the simulation is terminated.

multiplier: When a polony has a fractional coordinate bound that exceeds *trigger*, but not *abort*, it is time to rescale the mesh size. This is accomplished by adjusting each spatial coordinate mesh size by a factor of *multiplier*, and then, for each concentration variable, window-averaging the concentration values with *multiplier* as a window size.

Values of these dynamic rescaling parameters have been determined by experimentation and testing as described in section 3.4. In general, we use a *threshold* of 1e-4, a very conservative definition of the polony edge even in early cycles when concentration magnitudes are very small, and a *multiplier* of 2. *Trigger* and *abort* values differ from model to model.

Continuing with our 2D SPGM example above, in the first cycles, dx = .25μm and solution space covers a square from -20μm to 20μm in both x and y coordinates. Assume that *trigger* is .5. After enough cycles, the polony edge as defined by *threshold* will eventually move beyond 10μm from the origin and the *trigger* condition will be satisfied. At that point, the mesh size will be doubled so that solution space will cover a square from -40μm to 40μm in both x and y. The concentrations within the -20μm to 20μm square actually occupied by the polony are obtained by window-averaging the old values. Values in the -40μm to 40μm square outside of the interior -20μm to 20μm square are set to appropriate initial values (iP for P, eQ for Q, and 0 for all other variables).

Dynamic rescaling entails an intrinsic but small loss of accuracy due to the window averaging because, after rescaling, old concentration values must be represented on a mesh with half the resolution they had prior to rescaling. However, a more subtle effect arises in the 3D TPIM because window averaging of the 2D radial coordinate also leads to a loss of mass. This problem does not arise for Cartesian coordinates, and therefore does not arise in 2D simulations using dynamic rescaling or the 3D TPIM z-coordinate. The extent of this loss of mass is characterized in the following Theorem.

Theorem

Assume a molecule subject to radially symmetric 2D diffusion whose concentration *C* is at radial mesh point coordinates 0, 1, 2, … N is *C(0)*, *C(1)*, *C(2)* …, *C(N)*, for which the radial mesh size is *dr*. Assume as well that these *initial* mesh coordinates are subjected to rescaling as described above with a multiplier of 2, resulting in a new set of concentrations *C(0')*, *C(1')*, *C(2')* …, *C(N')*, where here the notation *C(k')* means the concentration of the molecule at *rescaled* mesh coordinate k. Then rescaling will result in a loss of mass of $((\pi * dr^2)/5)*(C(0)/4 + C(1))$.

Proof

The proof of this theorem depends on another proposition: Given a variable *C* that is subject to diffusion in a 2D radially symmetric solution space, mass must be calculated according to formula E3.3-1 in order that diffusion conserve mass:

(E3.3-1)
$$M = \pi \cdot dr^2 \left(\frac{C(0)}{4} + \sum_{k=1}^{N} 2k \cdot C(k) \right)$$

I give a proof of E3.3-1 in section 3.5 below. This result is entailed by the particular finite difference equations I have used to implement the diffusion equations. A different finite difference approximation would lead to a different formula.

One way of viewing E3.3-1 is to take it as defining the elements of area that must be used to integrate concentrations over the plane to calculate mass. The coefficient of the first term, $\pi * dr^2/4$, is the area of a circle of radius ½*dr*. Subsequent coefficients $2\pi*(k*dr)*dr$ are the finite difference approximations to the differential elements of area $2\pi r*dr$ that would be applied to radial integration of continuous radially symmetric functions over the plane. These specifications of element of area can be conveniently described as saying that the concentration *C(k)* at the radial mesh point with coordinate $k > 0$ is considered to be evenly spread over the interval k-½ to k+½ , while *C(0)*, the concentration at the origin, is considered to be spread over the half interval 0 to ½. This is diagrammed for both initial and rescaled coordinates in Figure F3.3-1 below. Ultimately, it is this different treatment of the origin compared to any other mesh point that leads to the loss of mass caused by rescaling.

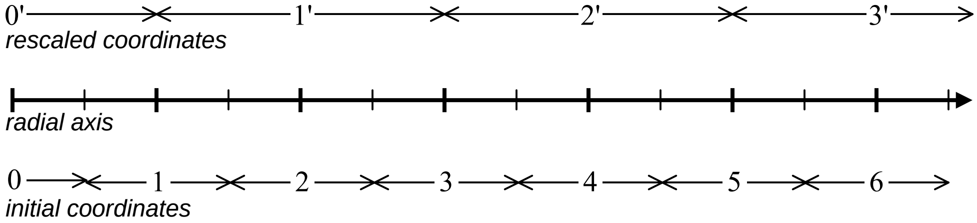


Figure F3.3-1: Division of radial axis into intervals corresponding to mesh point coordinates for both initial and subsequently rescaled coordinates. Thick vertical bars on radial axis correspond to intervals of length *dr*, the mesh size of the initial coordinates. Below, intervals corresponding to initial mesh points are centered on the initial coordinates (indicated by double arrows) and extend to the thin vertical lines on either side of the thick ones on the central axis, except for the interval about the origin. Above, rescaled coordinates and their intervals after rescaling is performed with *multiplier* 2.

This division of the radial axis into intervals shows how window-averaging must be performed during rescaling. Mesh point *k'* > 0 in the *rescaled* coordinates corresponds to mesh point 2*k* in the initial coordinates and is associated with an interval that extends from 2*k*-1 to 2*k*+1 on the initial scale. This includes the full interval surrounding initial coordinate 2*k*, but only ½ the intervals for 2*k*-1 and 2*k*+1, e.g. rescaled coordinate *1'* covers the full interval of initial coordinate *2* but only ½ of each of the intervals around initial coordinates *1* and *3*. The amount of material *C* in the interval around rescaled coordinate *k'* is therefore

$$\frac{1}{2}\cdot 2\pi\,((2k-1)dr)dr\cdot C(2k-1) + 2\pi\,(2kdr)dr\cdot C(2k) + \frac{1}{2}\cdot 2\pi\,((2k+1)dr)dr\cdot C(2k+1) =$$

$$2\pi\,dr^2\left(\frac{1}{2}\cdot(2k-1)\cdot C(2k-1) + 2k\cdot C(2k) + \frac{1}{2}\cdot(2k+1)\cdot C(2k+1)\right)$$

This material occupies an area of

$$\frac{1}{2}\cdot 2\pi\,((2k-1)dr)dr + 2\pi\,(2kdr)dr + \frac{1}{2}\cdot 2\pi\,((2k+1)dr)dr =$$

$$2\pi\,dr^2(4k)$$

Therefore the window-averaged concentration *C(k')* for rescaled coordinate *k'* is given by formula E3.3-2

$$\text{(E3.3 - 2)}\quad C(k') = \frac{1}{4k}\left(\frac{2k-1}{2}C(2k-1) + 2kC(2k) + \frac{2k+1}{2}C(2k+1)\right)$$

For the origin (*k'*=0), another formula applies. The interval around *0'* contains the complete interval for initial coordinate *0* and ½ the interval around initial coordinate *1*. The interval around *0'* therefore contains a mass of

$$\pi\frac{dr^2}{4}C(0) + \frac{1}{2}(2\pi\cdot 1dr)drC(1) = \pi\,dr^2\left(\frac{C(0)}{4} + C(1)\right)$$

in an area of $\pi dr^2(1/4 + 1) = 5\pi dr^2/4$, for the rescaled concentration given in E3.3-3:

$$\text{(E3.3 - 3)}\quad C(0') = \big(C(0) + 4C(1)\big)/5$$

These formulas describe the window-averaging that is required to compute the concentrations at rescaled coordinates. Now I will consider the total mass in the plane. For the initial coordinates I will call this quantity *M*. Its value, as computed using formula E3.3-1, is:

$$\begin{aligned}\text{(E3.3 - 4)}\quad M &= C(0)\cdot\pi\frac{dr^2}{4} + C(1)\cdot 2\pi\,(1\cdot dr)dr + C(2)\cdot 2\pi\,(2\cdot dr)dr + \ldots \\ &= \pi\, dr^2\left(\frac{C(0)}{4} + 2\cdot 1\cdot C(1) + 2\cdot 2\cdot C(2) + \ldots + 2\cdot n\cdot C(n) + \ldots\right)\end{aligned}$$

In rescaled coordinates, mass *M'* will be calculated by the same expression except for use of primed coordinates, where here dr' = the mesh size of the rescaled coordinates:

$$\begin{aligned}\text{(E3.3 - 5)}\quad M' &= C(0')\cdot\pi\frac{dr'^2}{4} + C(1')\cdot 2\pi\,(1\cdot dr')dr' + C(2')\cdot 2\pi\,(2\cdot dr')dr' + \ldots \\ &= \pi\, dr'^2\left(\frac{C(0')}{4} + 2\cdot 1\cdot C(1') + 2\cdot 2\cdot C(2') + \ldots + 2\cdot n\cdot C(n') + \ldots\right)\end{aligned}$$

However, here *dr'* = 2**dr*, and therefore $dr'^2 = 4dr^2$. Therefore, dividing E3.3-4 and E3.3-5 by πdr^2, we get:

$$\text{(E3.3 - 6)}\quad \frac{M}{\pi\cdot dr^2} = \frac{C(0)}{4} + 2\cdot 1\cdot C(1) + 2\cdot 2\cdot C(2) + \ldots + 2\cdot n\cdot C(n) + \ldots$$

and

$$\begin{aligned}\text{(E3.3 - 7)}\quad \frac{M'}{\pi\cdot dr^2} &= 4\left(\frac{C(0')}{4} + 2\cdot 1\cdot C(1') + 2\cdot 2\cdot C(2') + \ldots + 2\cdot n\cdot C(n') + \ldots\right) \\ &= C(0') + 8\cdot 1\cdot C(1') + 8\cdot 2\cdot C(2') + \ldots + 8\cdot n\cdot C(n') + \ldots\end{aligned}$$

The formulas E3.3-2 and E3.3-3 allow each of the *C(m')* terms in $M'/\pi dr^2$ to be expressed as a weighted sum *C(i)* terms for some set of *i*, and thereby allow comparison of $M/\pi dr^2$ and $M'/\pi dr^2$.

Consider first any initial mesh coordinate 2*k* for *k*>0 and consider the amount of *C(2k)* in *M'*. By E3.3-2, only *C(k')* will contribute *C(2k)* to *M'* , and *C(k')* contributes exactly 2*kC(2k)*/(4*k*) = ½ *C(2k)*. But by E3.3-7, *C(k')* has a coefficient of 8*k* in $M'/\pi dr^2$, so the coefficient of *C(2k)* in $M'/\pi dr^2$ is ½ 8*k* = 4*k* = 2*2*k*. But by E3.3-6 this is exactly the coefficient of *C(2k)* in $M/\pi dr^2$.

Now consider any initial mesh coordinate 2*k*+1 for *k*>0. The term *C(2k+1)* is contained in both *C(k')* and *C((k+1)')*. *C(k')* contains the term (1/(4*k*))((2*k*+1)/2)*C(2k+1)*, and has coefficient 8*k* in $M'/\pi dr^2$, for a total *C(k')* contribution of (2*k*+1)*C(2k+1)*. Likewise, *C((k+1)')* contains the term (1/(4(*k*+1))((2(*k*+1)-1)/2)*C(2(k+1)-1)* = (1/(4(*k*+1))((2*k*+1)/2)*C(2k+1)*, and has a coefficient of 8(*k*+1) in $M'/\pi dr^2$, for a total *C((k+1)')* contribution of (2*k*+1)*C(2k+1)*. Thus, the amount of *C(2k+1)* in $M'/\pi dr^2$ is ((2*k*+1) + (2*k*+1))*C(2k+1)* = 2(2*k*+1)*C(2k+1)*, and this is again exactly the amount of *C(2k+1)* in $M/\pi dr^2$.

Thus, $M'/\pi dr^2$ and $M/\pi dr^2$ contain identical amounts of *C(m)* for *m*>1, and the difference between them concerns only *C(0)* and *C(1)*. The only contribution of *C(0)* in $M'/\pi dr^2$ comes from *C(0')*, and by E3.3-3 the coefficient of *C(0)* in *C(0')* is 1/5. As the coefficient of *C(0')* in $M'/\pi dr^2$ is 1, there is (1/5)*1*C(0)* = *C(0)*/5 in $M'/\pi dr^2$. Meanwhile by E3.3-6, $M/\pi dr^2$ contains *C(0)*/4. Contributions to the *C(1)* term in $M'/\pi dr^2$ come from both *C(0')* and *C(1')*; specifically, *C(0')* contributes (4/5)*C(1)* and *C(1')* contributes *C(1)*/8, and these are weighted by coefficients of 1 and 8 respectively in $M'/\pi dr^2$, leading to a total of (1*(4/5) +8*(1/8))*C(1)* = (9/5)*C(1)* in $M'/\pi dr^2$. Meanwhile, there is 2*C(1)* in $M/\pi dr^2$ by E3.3-6. Therefore,

$$\frac{M}{\pi\, dr^2}-\frac{M'}{\pi\, dr^2}=\left(\frac{1}{4}-\frac{1}{5}\right)C(0)+\left(2-\frac{9}{5}\right)C(1)=\ \left(C(0)/4+C(1)\right)/5$$

and so

$$\text{(E3.3 - 8)}\quad M-M'=\frac{\pi\, dr^2}{5}\left(\frac{C(0)}{4}+C(1)\right)$$

QED.

Immediate consequences of this result are:

- $M \geq M'$. I.e., mass is never gained during rescaling, but can only be lost.
- Rescaling can only cause loss of mass at mesh points *0'* and *1'*, i.e., loss at the origin and one mesh point away, and never further away.
- The amount of mass loss is O(dr^2) and so can be reduced by reducing *dr*. However, an important exception to this is when *C* is a delta function at the origin, so that reducing *dr* results in a compensating increase in *C(0)*. In such a case, *C(0)* = $4/(\pi dr^2)$ and *C(k)* = 0 for *k*>0 (this integrates to 1 under formula E3.3-1). Rescaling under these circumstances leads to *M'* = 4/5 by E3.3-8.

This last result can easily arise in 3D TPIM simulations. If, on simulation initialization, ST is assigned a delta function, it immediately dissociates into a molecule of S and a molecule at T at the beginning of the first denaturing cycle. S then diffuses, but T, being tethered, does not diffuse and remains a delta function. If the solution space is small, S may diffuse far enough to trigger a rescaling operation. The resulting loss of mass may not be large for S, as by diffusion much of its mass will have drifted away from the origin during the denaturing cycle. But T remains a delta function and undergoes the mass loss just described, resulting in a mass of 0.8 after rescaling. It was observation of this mass loss effect, and the difficulty of reducing it by reducing *dr*, that led us to work out the mathematics above.

We compared mass differences computed by the E3.3-8 for S1d before and after rescaling from an actual 3D TPIM simulation and found the differences between *M-M'* computed from E3.3-8 and directly to be the same up to ~1% error (0.0005948 via E3.3-8 vs. 0.0005883 by direct computation). As the formula should be exact, a 1% error is surprising, but the direct computation involves a large number of calculations with small numbers and the error is likely a result of round-off error. (Reference file: outdata.dr_1.rb100.dz1.zb10.cyc2.R.rrs.S1d.S1s.check.xls)

The three main ways of reducing the problem of mass loss at the origin are:

(a) Keep reducing *dr*. Although as noted above, reducing *dr* does not reduce mass loss at initialization time so long as ST remains a delta function, once *dr* becomes sufficiently small a pure delta function initial condition for ST becomes impossible and mass must be spread into a region around the origin (see section 3.1). If *dr* is small enough, very little mass remains at the origin itself and mass loss becomes insignificant.

(b) For a given *dr*, increase the number of mesh points by increasing *rb*, effectively increasing the size of solution space. This gives the growing polony more room and effectively allows rescaling to be delayed until the polony is larger. Mass loss will still affect the origin when rescaling eventually occurs, but as a larger polony has more mass distributed away from the origin than a smaller one, and rescaling does not affect such mass, a larger amount of mass is correctly conserved.

(c) Increasing the *trigger* parameter for rescaling also delays rescaling and has a similar effect to (b).

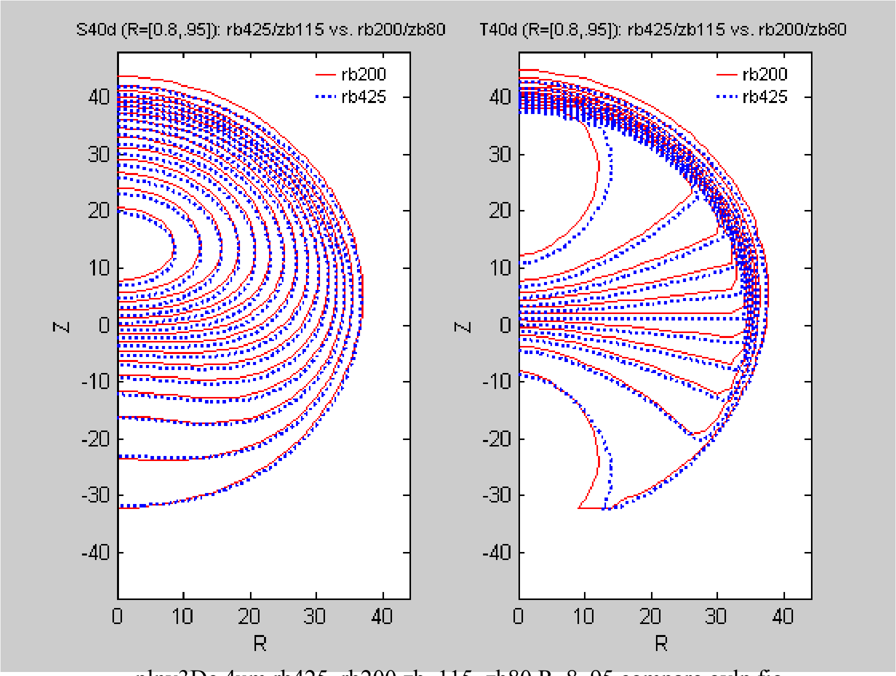
3.4 Simulation and robustness of results computed with dynamic rescaling

As noted in section 3.3, we developed dynamic rescaling to make simulations based on polony models using multiple spatial coordinates possible in practical amounts of computer time. However, as also noted in section 3.3, dynamic rescaling can introduce inaccuracies into simulation results, most notably a loss of mass in 3D TPIM simulations. We therefore assessed the impact of rescaling on simulation results on several features of polonies described in the main article. The observations presented here show that the impact of rescaling is generally small.

Generally, our method for assessing the impact of rescaling has been a variant of *Test C* of section 3.2, defined there as a comparison of results from two simulations that are the same except for having different mesh sizes. The variant applied to dynamic rescaling simulations is to compare the results of simulations that have the same mesh sizes but different solution space sizes. The reason this works is that in dynamic rescaling simulations mesh sizes are increased when the polony grows sufficiently close to the solution space boundary; therefore, mesh sizes don’t increase as often in simulations with larger solution spaces, maintaining them at finer average spatial resolution over the course of the simulation. These simulations will also tend to undergo fewer rescalings, exposing them less to any biases introduced by rescaling, and they will undergo their first rescalings at later cycles, reducing any amplification of any biases introduced early in the simulation when the polonies are very small.

3.4.1 3D TPIM polony interaction geometry

The complex geometry of polony invasion is described in the main article. In 3D TPIM simulations (e.g., Figure 6 of main article), invasion contours indicate less invasion density near the z-axis vs. away from it. As one of the biases of dynamic rescaling is a loss of mass near the z-axis, the possibility exists that this apparent feature of 3D TPIM invasion could be the result of a artifact introduced by the dynamic rescaling procedure. Therefore we explored the degree to which this geometry might depend on dynamic rescaling. The close conservation of the geometry in two simulations, one of which used a larger solution space, is seen in Figure F3.4.1-1, and suggests that these results are not artifactual. That similar geometry is seen in 2D TPIM simulations (Figure 7 of main article) which use only Cartesian coordinates and therefore are not subject to loss of mass under rescaling, is additional evidence for this conclusion. The simulation with the larger solution space features slightly steeper gradients compared to the one with the smaller solution space, with contours compressed into a slightly smaller area, and appears to shift the polony very slightly towards Z=0.



plny3De.4um.rb425_rb200.zb_115_zb80.R_8_95.compare.ovlp.fig

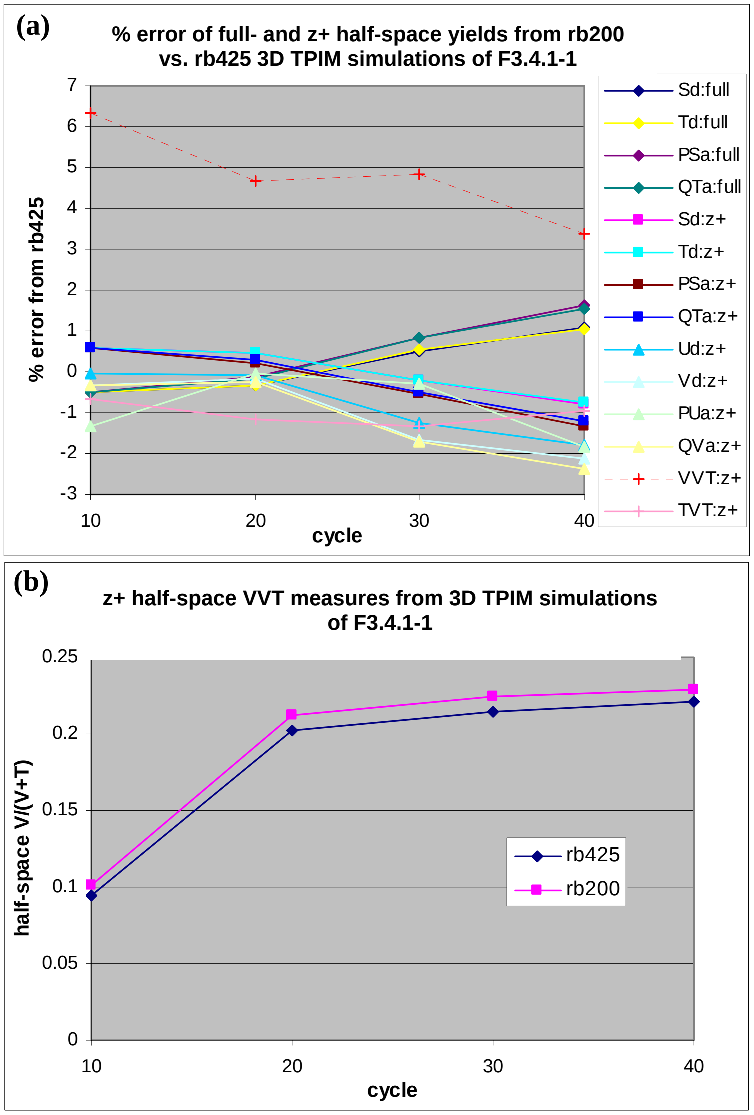
Figure F3.4.1-1: Geometry of polony invasion for two polony seeds at +/- 4μm on the z-axis, as shown by contour maps of S and T after denaturing cycle 40 (S40d and T40d), as shown in two simulations, one with a solution space of 201 radial and 161 z-axis mesh points (red, rb200/zb80) and the other with 426 radial and 231 z-axis mesh points (blue dashed, rb425/zb115). The rb425/zb115 simulation ran with an average mesh size ~ 2x finer than the rb200/zb80 simulation, and underwent two rescalings (cycles 12 and 36) *vs.* three rescalings (cycles 4, 11, 24).

3.4.2 3D TPIM measures of polony yield and exclusion

Loss of mass during dynamic rescaling in 3D TPIM simulations may be viewed as a penalty paid for accurate preservation of concentrations. Since the equations of polony models feature only concentrations and not masses, and rescaling preserves these accurately, the close correspondence between concentration profiles seen in the two simulations compared in Figure F3.4.1-1 may not be surprising, and we might expect to see more divergence when comparing yields. We therefore compared yields of molecular species computed by the two simulations described in F3.4.1-1 at various cycles. The results are shown in Figure F3.4.2-1.

Total masses, computed over the full 3D solution space, show a low degree of relative error that rises over the course of the simulations (Figure F3.4.2-1a, diamond markers). Shown are relative errors both for not only the strand species S and T generally taken to indicate polony yield, but also the generative species PS and QT found in growth faces after annealing phases (see main article). The relative error exhibited by these full-space yields generally rises over the course of the simulations and reaches a maximum relative error in cycle 40 that is < 1.7%. The rise in relative error indicates that the coarser resolution simulation (rb200) exhibits a small but increasing tendency to overestimate yields compared to the finer resolution simulation (rb425)

As described in the main article, half-space yields are useful numerical measures of the degree of polony infiltration and are used to compute the VVT measure of polony invasion. In the simulation of F3.4.1-1, the T polony 'owns' the positive half space z>=0, here denoted 'z+', and the V polony owns the negative half space (z<=0). Figure F3.4.2-1a shows that the relative error for half-space yields tends to exhibit the opposite trend compared to full-space yields: the coarser resolution simulation (rb200) shows an increasing tendency to underestimate them compared to the higher resolution one (rb425). Half-space yields of species for the polony that owns the half-space (Figure F3.4.2-1a, square markers) tend to be underestimated less than those for invading species (Figure F3.4.2-1a, triangular markers). By cycle 40 the greatest absolute relative error for any of these half-space yields is modest at < 2.4% (QV40a: -2.39%). These relative error computations are quite sensitive to how the values at z=0 are figured in (see F3.4.2-1 legend for definition of z+ half-space yield).



plny3De.4um.rb200_zb80.rb425_zb115.R_8_95.compare.xls

Figure F3.4.2-1: Precision of yields computed from 3D TPIM simulations compared in Figure F3.4.1-1 (rb425 and zb115 vs. rb200 and zb80).

(a) % error for yields or yield ratios for various molecular species every 10 cycles over the full 3D solution space (full) or the z>=0 half-space (z+). The % error for X is

$$100(X_{rb200} - X_{rb425})/X_{rb425}$$

The z+ half-space yield for C is

$$2\pi\left(\sum_{z=0}^{zb}\gamma(z)\left(\left(\sum_{r=1}^{rb} r\cdot C(r,z)\right)+\frac{C(0,z)}{8}\right)\right)dr^2dz$$

where γ(z)=1/2 for z=0 and 1 otherwise (also see section 3.5). Sd, Td, Ud, Vd = species yields after denaturing phase; PUa, PUa, QTa, QVa = species yields after annealing phase; VVT = Vd/(Vd+Td); TVT = Td/(Vd+Td); diamond markers = full space yields; square markers = z+ yields for species associated with the T polony; triangular markers = z+ yields associated with the V polony.

(b) V/(V+T) in the z+ half space after every 10 denaturing cycles for the two simulations. The % error between corresponding V/(V+T) values is the VVT:z+ line in (a) (dashed line).

Figure 3.4.2-1b compares the z+ half-space VVT measure for these two rescaling simulations. The relative error for this quantity, shown in Figure 3.4.2-1a (dashed line, VVT:z+), is higher than for any individual half-space yield, although it appears to be declining as cycles progress. Despite the relatively high degree of error, however, the VVT graphs for the two simulations in F3.4.2-1b show a very orderly relation. As also indicated by the consistently positive relative error in F3.4.2-1a, F3.4.2-1b shows that the coarser resolution simulation (rb200) consistently overestimates the amount of invasion by a neighboring polony compared to the finer resolution simulation (rb425), but not by much.

3.4.3 2D SPGM yields

We also examined the accuracy of the two 2D SPGM simulations used for the SPGM yield analysis described in Figure 4b of the main article. The two SPGM simulations are identical except that one considers the polonies to be growing in a 1μm thick 2D gel while the other considers them to be growing in a 10μm thick 2D gel. To assess the accuracy of these simulations, each was run twice, one with x and y spanning mesh points with coordinates between –80 and 80 (xb80), and the other with these variables spanning mesh points between –160 and 160 (xb160). Both used an initial mesh size dx = .25μm. Information on the simulations is given in Table T3.4.3-1. These were 100 cycle simulations and therefore ran ~2.5x as long as standard 40 cycle simulations.

thickness↓ \ resolution→	xb80	xb160
1μm	hh:mm:ss = 10:39:07 rescalings: 5, 13, 28, 56	hh:mm:ss = 43:23:30 rescalings: 12,28,57
10μm	hh:mm:ss = 10:40:10 rescalings: 5, 13, 28, 55	hh:mm:ss = 43:24:04 rescalings: 12,28,57

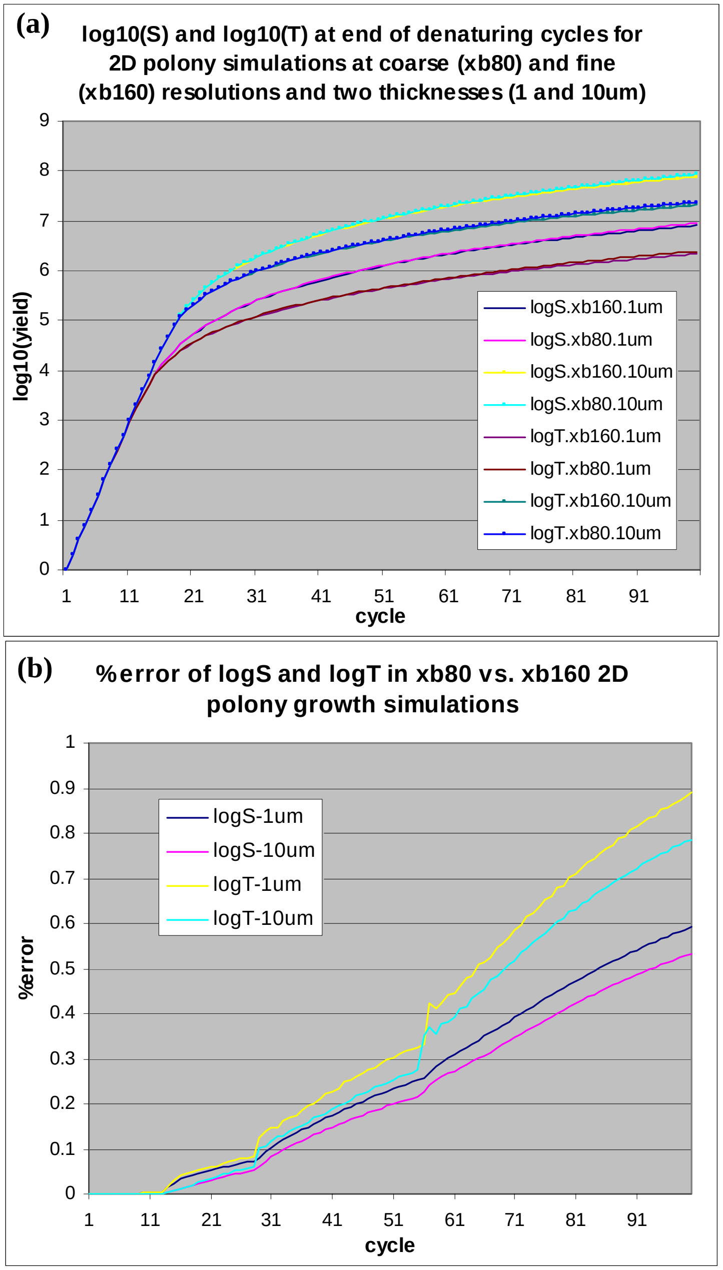
Table T3.4.3-1: Run times (hh:mm:ss) and cycles in which rescalings occurred in 2D SPGM performed on 1Ghz processors.

Results of the comparison of corresponding xb80 and xb160 simulations are shown in Figure F3.4.3-1. It is clear that each rescaling operation results in a jump and subsequent increase in the amount of divergence between the two simulations. However, even at cycle 100, the divergence between yields when assessed on a log10 scale is very small; it is more sizeable but still modest when yields are assessed directly (with a maximum deviation of ~14%). Similar to what was observed for yields in the 3D TPIM comparison in Figure F3.4.2-1a, the coarser mesh resolution consistently overestimates yields compared to the finer mesh resolution.

3.4.4 2D polony yield power-law regressions

We also performed the windowed power-law regressions (see main article and legend for F3.4.4-1) specifically shown in Figure 4b of the main article for the xb80 and xb160 2D SPGM simulations described in Table T3.4.3-1. The results are seen in Figure F3.4.4-1a, where it can be seen that at xb80 resolution the power-law exponent (B) is visibly affected by rescaling operations. At xb160, the effects are much less. Also of note is that the other parameters solved by the regression (A and C) are even more sensitive to rescalings than the exponent (B) (see Figure F3.4.4-1b).

Because of the sensitivity exhibited by the xb80 results, the xb160 results were used in the main article. However, we note that these were non-standard 100 PCR cycle simulations that were run specifically to discover long term yield trends beyond the standard 40 PCR cycles. If we limited our analysis to 40 cycles, the xb80 and xb160 yield results would be equivalent.



polony2D.dx_25.xb80.xb160.c100.2.outtotals.compare.xls

Figure F3.4.3-1: Sensitivity of 2D polony growth yields to rescaling parameters

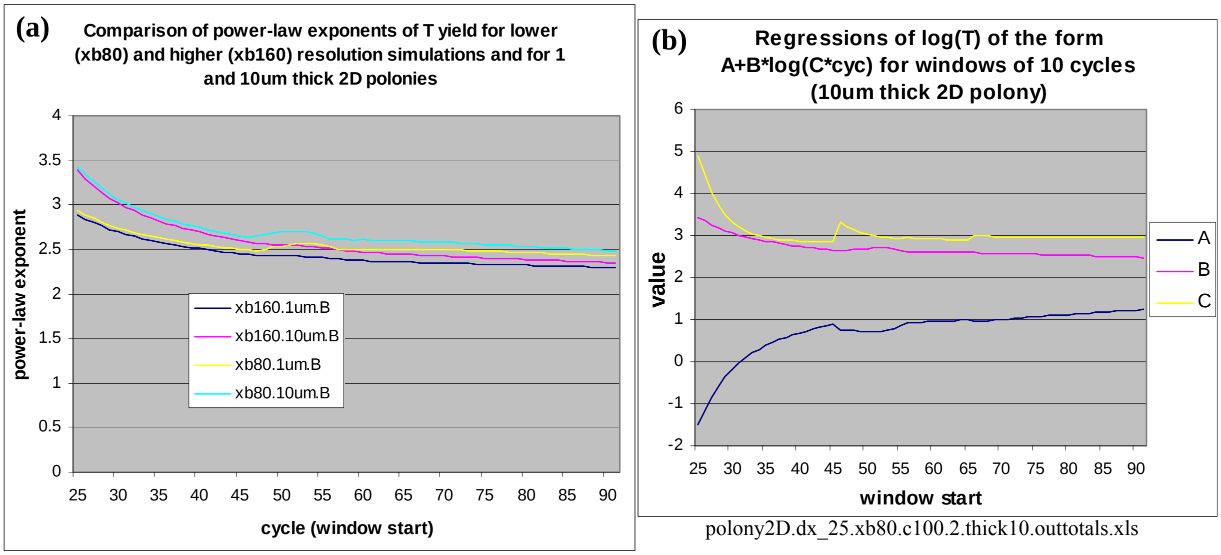
(a) $\log_{10}(S)$ and $\log_{10}(T)$ for the four 2D SPGM simulations described in Table T3.4.3-1. The yield curves of corresponding variables in the xb80 and xb160 simulations are superimposed, indicating good precision at the coarser xb80 resolution.

(b) % error of the xb80 log10(S) and log10(T) values with respect to the same values in the finer mesh xb160 simulations. % error of X = $(X_{xb80} - X_{xb160})/X_{xb160} \cdot 100$. As can be seen, % error jumps at each of the cycles during which there was a rescaling (see Table 1), indicating that the two simulations diverge when a rescaling operation takes place. After %error jumps, its curve then also shows an increased slope. T values exhibit larger % error than S values. % error values shown here are all very small and < 1% by cycle 100. However, these values are computed from $\log_{10}$s of S and T. When % error is computed directly from S and T values, % error of S and T at cycle 100 ≈ 10% and 14%, respectively, and ≈ 2.3% and 2.8% at cycle 40.

3.4.5 2D polony geometry

Figure F3.4.5-1 shows contour maps of S after denaturing in cycle 40 and in cycle 100 for the two 2D SPGM simulations described in Table T3.4.3-1. Consistent with observations made in section

3.4.4 above, there is much less divergence of polony geometry at cycle 40 than there is at cycle 100, again indicating that standard 40 cycle simulations at xb80 exhibit good precision at this coarser mesh size.



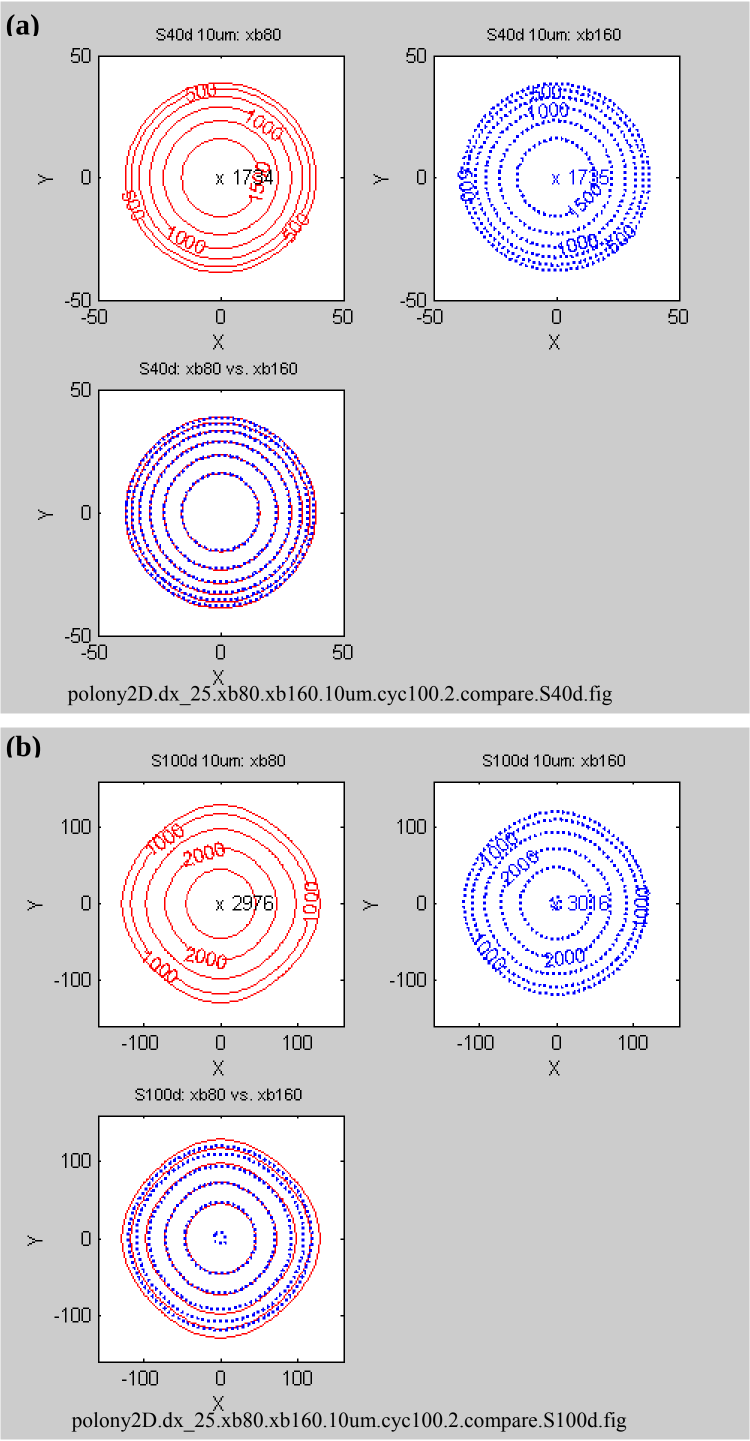
polony2D.dx_25.xb80.c100.2.thick10.outtotals.xls

polony2D.dx_25.xb80.xb160.c100.2.outtotals.compare.xls

Figure F3.4.4-1: Effect of rescaling on power-law regressions of $\log_{10}(T)$ yield curves for the 2D SPGM simulations described in Table T3.4.3-1. (a) Power-law exponents derived as B solution values to regressions of the form $\log_{10}(T) = A+B*\log(C*cyc)$ performed over windows of 10 cycles. The xb160.1um.B and xb160.10um.B curves are identical to the curves in Figure 4b of the main article but are here supplemented with the corresponding xb80 curves. The xb80 curves show an upward kink starting around cycle 45 and stretching for ~10 cycles that is a result of the rescaling operation at cycle 55. As regressions are done in windows of 10 cycles, the effect of the rescaling is first taken into account in the window starting at 45. The 'kink' also offsets the B curves upward compared to xb160 B values, after which they resume their decline. No such 'kink' is visible in the xb160 B curves in the same region despite a rescaling at cycle 57. (b) Example of a regression on one series (xb80, 10μm gel) showing that the other regression parameters A and C are more affected by rescaling than B.

3.4.6 Possible source of inaccuracy in 2D SPGM simulations

The 2D SPGM simulations described in Table T3.4.3-1 are performed in a model that employs only Cartesian coordinates and are therefore not subject to the loss-of-mass bias characterized for 3D TPIM simulations (section 3.3). The question therefore arises: What causes the divergence between the 2D SPGM xb80 and xb160 results. One answer may be that the divergence merely affects the increased accuracy of the (on average) finer mesh used by the xb160 simulation. But while this may indeed account for some divergence between xb160 and xb80 results, Figures F3.4.3-1 and F3.4.4-1 indicate that divergence is specifically introduced by rescaling operations. One possible cause of this is the effect of rescaling on the distribution of Q (free primer).



polony2D.dx_25.xb80.xb160.10um.cyc100.2.compare.S40d.fig

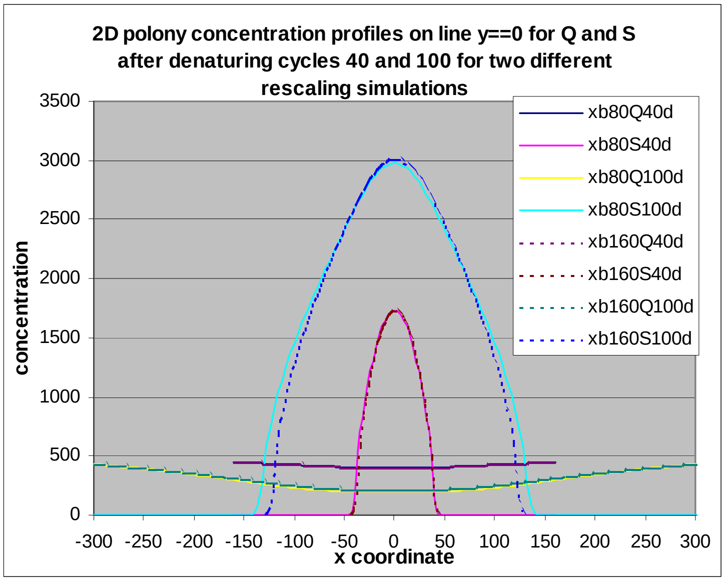
polony2D.dx_25.xb80.xb160.10um.cyc100.2.compare.S100d.fig

Figure F3.4.5-1: Effect of rescaling on 2D polony geometry. (a) Superimposition of contour maps of S after denaturing in cycle 40 for the xb80 and xb160 2D SPGM simulations described in Table T3.4.3-1. (b) Superimposition of contour maps for the same simulations for S after denaturing cycle 100.

Contours superimpose better at cycle 40 than they do at cycle 100, consistent with other observations that divergence increases with more cycles. Another interpretation is that contours in (b) within the spatial region depicted in (a) show better superimposition than contours outside of the region, which developed after cycle 40. (Note the different scales in (a) vs. (b) reflecting the larger polony size in (b)).

Q is not examined by rescaling logic, only S (and sometimes U) (see section 3.3). This is done for practical reasons. S (and, where applicable, U) increases sharply from negligible values at the edge of a polony, making it easy to use S to determine where the polony edge is and, by this means, consistently and predictably control the times at which rescaling operations occur. By contrast, in 2D simulations, Q exhibits a bowl-shaped concentration profile that is high at the edge of solution space (where there is a source of Q) and which is only gradually diminished as one moves towards the center of the space (see Figure F3.4.6-1). This makes it difficult to define a threshold that can consistently and predictably be used to trigger rescaling.

But, by this same token, rescaling has a more substantial impact on Q than on other polony molecules. Except for P, rescaling affects all molecules other than Q by moving a solution space boundary at which the molecules have zero concentrations through a large region of surrounding space that is also empty of these molecules. P, by contrast, is at an initial constant non-zero concentration at the solution space boundary, and this boundary is extended to include a larger region of space with this same non-zero concentration of P. In either case, for all molecules except Q, rescaling should have no direct effect on the concentrations or dynamics of these molecules. But rescaling has a direct impact on Q because it effectively removes the Q source at the solution space boundary to a greater distance away from the polony. This changes the dynamics of the way the Q source feeds the polony, thereby indirectly affecting further polony evolution (Figure F3.4.6-1). (*How* it should affect the polony is harder to predict. In a smaller solution space, more frequent rescaling will more frequently remove the Q source away from the polony, but by the same token, the polony will more frequently come closer to the source.)



polony2D.dx_25.xb80.xb160.2.thick10.Q40Q100.compare.xls

Figure F3.4.6-1: Q and S concentration profiles at cycles 40 and 100 of the xb80 and xb160 10µm 2D SPGM simulations (Table T3.4.3-1). Shown are the concentrations along the y axis of the 2D solution space. Unlike S and other polony molecules, Q has a gradually changing concentration at the edge of solution space. The cycle 40 Q curves stop at +/- 160 vs. +/- 300 for the cycle 100 curves because solution space was confined within +/- 160 until a rescaling operation extended it. At the time of that rescaling, the Q source that used to be at +/- 160 was removed to +/- 300, directly affecting Q dynamics and indirectly affecting subsequent polony evolution, especially near the polony edge. Note that S100 profiles differ in the xb80 vs. xb160 simulations mainly at the edge.

If rescaling has an impact on simulation results due to its effect on Q concentrations, it may not be accurate to consider this a "distortion" imposed by rescaling, because the identification of the solution space boundary as a source of Q is itself an idealization of what goes on in actual polony gels. In reality there is no Q source that maintains a constant concentration of Q in the gel, only a large reservoir of Q that diffuses into the polony. Specification of a source at the boundary is only a convenience that avoids the computational overhead of defining the huge solution space that would be needed to accurately represent this reservoir. In this sense, rescaling may actually help represent polony growth more accurately, but only by introducing another artifact – a gradually receding source at the edge of an ever-growing reservoir.

3.5 Calculation of polony yield

This section shows how the formula used to compute yields in a polony model that uses a radial coordinate depends on the implicit equations used to solve PDEs in that model. Discussion is restricted to the 2D radial coordinate used in 3D TPIMs.

In 2D where there is radial symmetry, the diffusion equation has the form:

$$\text{(E3.5 - 1)} \quad \frac{\partial C}{\partial t} = D \cdot \frac{1}{r} \frac{\partial (r \cdot \frac{\partial C}{\partial r})}{\partial r} = D \cdot \frac{\partial^2 C}{\partial r^2} + \frac{D}{r} \frac{\partial C}{\partial r}$$

where *C* is the concentration of the diffusing substance and *D* is its diffusion coefficient. In the following we will assume that *D* is a constant > 0, and that *C* satisfies regularity assumptions adequate for the argument in E3.5-2 below (specifically, $\partial C/\partial r$ is finite at the origin, $r \cdot \partial C/\partial r \to 0$ as $r \to \infty$, and *C* satisfies whatever continuity and differentiability requirements are needed to allow the partial derivative with respect to time to be moved behind the integral sign in E3.5-2b).

Equation E3.5-1, plus the additional assumptions mentioned, guarantee that the total mass *M* of *C* is conserved. This is easily shown by the following argument:

$$\text{(E3.5 - 2a)} \quad M = \int_0^\infty 2\pi\, rC(r)dr$$

$$\text{(E3.5 - 2b)} \quad \frac{\partial M}{\partial t} = \frac{\partial \int_0^\infty 2\pi\, rC(r)dr}{\partial t} = \int_0^\infty 2\pi\, r \frac{\partial C(r)}{\partial t} dr = \int_0^\infty 2\pi\, rD \cdot \left(\frac{1}{r} \frac{\partial (r \frac{\partial C(r)}{\partial r})}{\partial r} \right) dr \quad \text{[by E3.5 - 1]}$$

$$\text{(E3.5 - 2c)} \quad \frac{\partial M}{\partial t} = 2\pi\, D \int_0^\infty \frac{\partial (r \frac{\partial C(r)}{\partial r})}{\partial r} dr = 2\pi\, D \left[r \frac{\partial C(r)}{\partial r} \right]_0^\infty = 0 \quad \text{[by regularity assumptions]}$$

This argument is transferable to the approximate solution of diffusion equations using finite difference equations. Explicit versions of the finite difference equations used in the 2D radial coordinate of the 3D TPIM model are given in E3.5-3 (Crank, 1956, equations 10.41 and 10.42). Here *C(m,t)* represents the concentration at spatial mesh point m at time mesh point *t*, where the radial mesh distance is Δ*r* and the temporal mesh distance is Δ*t*. (Therefore the actual radial distance of mesh point *m* is *m*Δ*r*, and similarly for *t*.) There are two difference equations, one for *m*>0 and the other for *m*=0. The boundary condition assumed for this finite difference approximation is that *C(N,t)* = 0

$$\text{(E3.5} - 3) \quad C(m,t+1) - C(m,t) = D\frac{\Delta t}{2m\Delta r^2}\big((2m+1)C(m+1,t) - 4mC(m,t) + (2m-1)C(m-1,t)\big) \ (m > 0)$$

$$C(0,t+1) - C(0,t) = 4D\frac{\Delta t}{\Delta r^2}\big(C(1,t) - C(0,t)\big)$$

The finite distance analogue to E3.5-2a is:

$$\text{(E3.5} - 4) \quad S(t) = \sum_{m=0}^{N} 2\pi\,(m\Delta r)C(m,t)\Delta r$$

Notice that in E3.5-4, the term for *m*=0 vanishes and there is no contribution by *C(0,t)*. Therefore the summation may just as well begin at *m*=1.

Proceeding with the argument analogous to the rest of E3.5-2 does not generate the identity *S(t+1)* - *S(t)* = 0, but rather another expression:

$$\begin{aligned}
\text{(E3.5 - 5)} \quad S(t+1) - S(t) &= 2\pi\Delta r^2 \sum_{m=1}^{N} m(C(m,t+1) - C(m,t)) \\
&= 2\pi\Delta r^2 D\frac{\Delta t}{\Delta r^2}\sum_{m=1}^{N} m\left(\frac{1}{2m}\big((2m+1)C(m+1,t) - 4mC(m) + (2m-1)C(m-1,t)\big)\right) \\
&= \pi D\Delta t\sum_{m=1}^{N}\big((2m+1)C(m+1,t) - 4mC(m,t) + (2m-1)C(m-1,t)\big)
\end{aligned}$$

In this sum, all terms *C(k,t)* cancel out for *N*>*k*>1. Specifically, such a term appears three times, once for *m* = *k*-1, where it gets a coefficient of (2(*k*-1)+1) = 2*k*-1, once for *m* = *k*, where it has a coefficient of –4*k*, and once for *m* = *k*+1, where it gets a coefficient of (2(*k*+1)-1) = 2*k*+1; the sum of these three coefficients is 0. The term *C(N,t)* vanishes because *C* is 0 at the boundary radius by our assumed boundary condition. Therefore, *S(t+1)-S(t)* contains only terms *C(0,t)* and *C(1,t)*. *C(0,t)* only appears in the term for *m*=1 as (2*m*-1)*C(m-1,t)* = *C(0,t)*. *C(1,t)* appears in two terms: In the *m*=1 term it appears as -4*C(1,t)*, and in the *m*=2 term it appears as (2*m*-1)*C(m-1,t)* = 3*C(1,t)*. Thus, *C(1,t)* appears in *S(t+1)-S(t)* as (-4+3)*C(1,t)* = -*C(1,t)* and:

$$\text{(E3.5 - 6)} \quad S(t+1) - S(t) = \pi\, D\Delta t\big(C(0,t) - C(1,t)\big)$$

But by the finite difference equation for *m* = 0 in E3.5-3, this becomes

$$\text{(E3.5} - 7) \quad S(t+1) - S(t) = -\frac{\pi\,\Delta r^2}{4}\big(C(0,t+1) - C(0,t)\big)$$

Thus, *S(t)*, the direct analogue of *M* in E3.5-2a, is not conserved over time. However, if we define *M(t)* as *S(t)* + π(Δr^2/4)*C(0,t)*, E3.5-7 says that *M(t) is* conserved over time. Therefore the definition of mass that is conserved by difference equations E3.5-3 is

$$M(t) = \frac{\pi\,\Delta r^2}{4}C(0,t) + 2\pi\,\Delta r^2\sum_{m=1}^{N} mC(m,t)$$

QED

I note two additional points:

- The difference equations actually used for the 2D radial coordinate of the 3D TPIM are Crank-Nicolson variants of E3.5-3 in which all *C(k,t)* terms on the right sides of the equations are replaced by (*C(k,t)*+*C(k,t+1)*)/2 (see E3.6.1-1). However, the argument above works just as well with these more complicated equations and results in the same conclusion.

- Notice that the result of this proof is that conservation is achieved by adding back a contribution for the *C(0,t)* term that vanished from *S(t)* itself. A similar argument, when applied to the finite difference equations used for the 3D SPGM (polony3D), shows that the 3D analogue of *S(t)* is directly conserved. This has the strange consequence that in the 3D SPGM, concentrations at the origin are *not* counted towards the total mass of a molecular species, while these masses are included in the 3D TPIM!

We wish to acknowledge M. Bennett for calling our attention to the argument for conservation of mass in continuous PDEs in equations E3.5-2 and pointing out that the finite difference equivalent must be used for yield calculations.

4 Citing this document

Please cite the main article when referring to this supplemental information document:

John Aach and George M. Church, 2004, Mathematical models of diffusion-constrained polymerase chain reactions: basis of high-throughput nucleic acid assays and simple self-organizing systems, *J. Theor. Biol* 228:31-46

5 References

Ames, W.F., 1992. Numerical methods for partial differential equations. San Diego, Academic Press

Berg, H.C., 1993. Random walks in biology. Princeton University Press, Princeton

Crank, J., 1956. Mathematics of diffusion. Oxford University Press, London

Mitra, R.D., Church, G.M., 1999. In situ localized amplification and contact replication of many individual DNA molecules. Nucleic Acids Res, 27 (24), e34.

Press, W.H., Flannery, B.P., Teukolsky, S.A., et al., 1996. Numerical Recipes in C. Cambridge University Press, Cambridge