Intercellular interaction mechanisms for the origination of blast crisis in CML

Supplementary Materials.

These supplementary materials cover the following: robustness of our results and conclusions with respect to variations of the stem cell proliferation parameter (Section S1); stochastic vs. determinisitic models (S2); the statistical methods used and response to changes in those methods (S3); response to changes in the blast crisis latency period L_BC (S4) or to the ratio α/β that determines the ratio of cycling to quiescent stem cells (S5); the source code we used (S6); and the supplement's bibliography (S7). Parameter values are those of Table 1 in the main text unless explcitly indicated otherwise.

S1. Robustness relative to changes in the stem cell proliferation parameter

An important parameter for almost all mathematical CML models is the value or probability density distribution for the CML latency time. The estimates for this parameter are based mainly on data for extra CML cases induced by exposure to ionizing radiation, which can cause BCR-ABL translocations [Radivoyevitch et al. 2001]. Most current CML models, perhaps following the supplementary material for [Michor et al. 2005], estimate that this latency time distribution has one fairly sharp peak at approximately 5-10 years; this estimate is based on analyses of data for leukemias in the Japanese atomic bomb survivors (e.g. [Preston et al. 1994]), sometimes combined with data on iatrogenic cancers after radiotherapy (e.g. [Radivoyevitch & Hoel 2000]). Both the estimate itself and its application to the sporadic leukemias of interest here are problematical. As regards the second point, sporadic CML, perhaps unlike radiogenic excess CML at doses ~2 Gy, is presumably initiated by a single Ph+ clone. Shortly after initiation such a clone is far too small to be actually detected *in vivo* so the time of initiation is not observable; in addition, stochastic extinction of such a clone even if its cells have a growth advantage is a possibility. As regards the estimate itself, the radiation data is under frequent revision. The minimum latency for leukemias could be considerably shorter than five years [Little et al. 1999; Little et al. 2008]; and a recent analysis of more extensive versions of the atomic bomb data specifically for CML suggests that the time distribution, instead of being concentrated in a comparatively narrow interval, may be spread out over as much as five decades [Richardson et al. 2009].

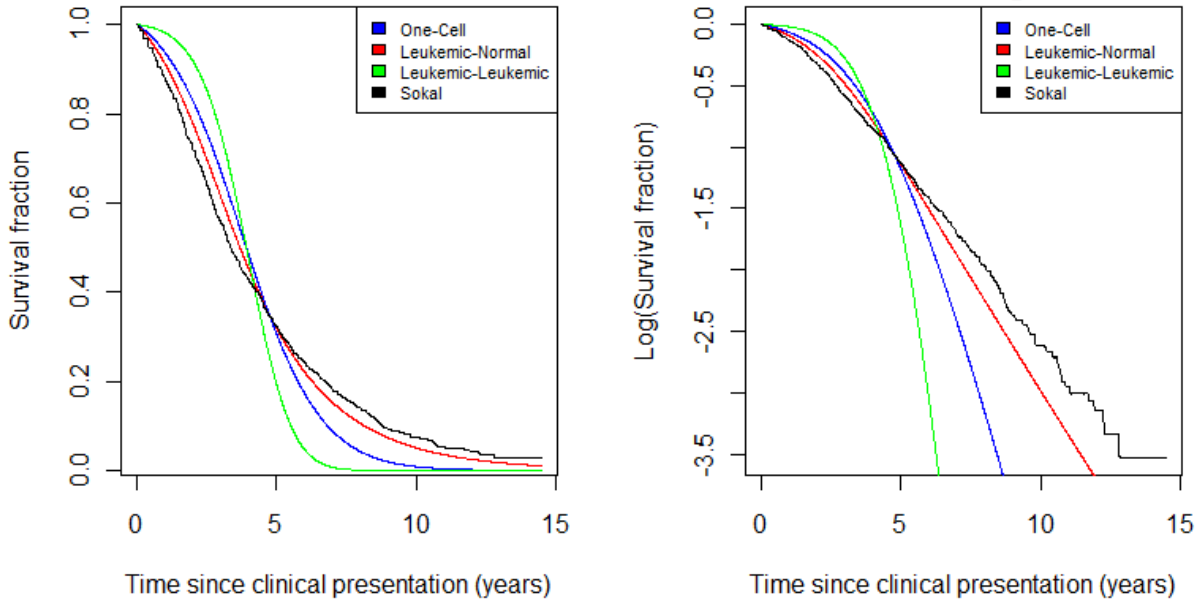
In the main text we denoted the CML latency time (characterized in Figure 1B) by the parameter *L_CML* and computed it as explained in the Equations and Assumptions Section. The primary parameter going into the calculation of *L_CML* was the proliferation parameter *r* for the specific case of untreated leukemic cycling stem cells (Table 1), denoted by *r′* in what follows; in the original estimates *L_CML* was used to fix *r′*. Because of the problems mentioned above we next consider various possible values for *r′*, calculate the resulting values of *L_CML*, repeat the calculations of the main text for blast crises, and estimate the robustness of our overall conclusions with respect to variations of *r′*.

The figures below are similar to Figure 4 of the main text but for visual clarity we do not include all 45 curves, only 3 curves, one best fit for each color; black curves are for the data. The p-value tables below contain 45 entries instead of 18 but otherwise are similar to Table 2 of the main text; they include the full table from which Table 2 was constructed. "NLLS" refers to use of non-linear regression in the calculation of λ, as discussed in section S3. Tables labeled "lambda" give the values of the calculated blast crisis origination intensity parameter λ with units $(\text{day})^{-1}$.

Sachs et al. Supplementary Materials. S1. Robustness relative to changes in the stem cell proliferation parameter

Parameters used: those of Table 1 except that r'=0.0065, which implies L_CML=17.8 years.

Representative best fits: red = (y0, x1), green = (y0, y0), blue = (y0, 1)

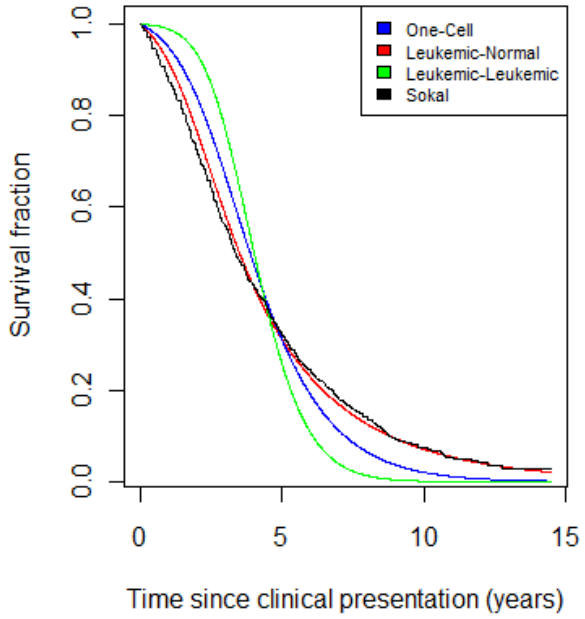


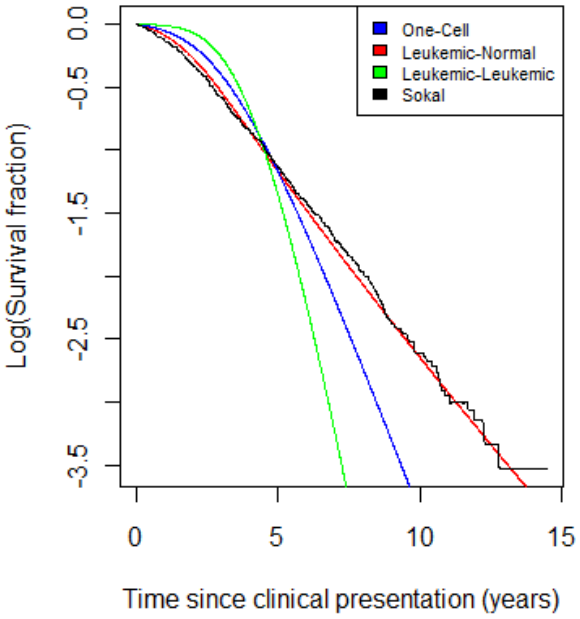

```
> pval_NLLS
          1             xq             x0             x1             x2             x3
yq   2.265216e-12 1.248653e-11 3.244479e-06 5.354481e-06 5.798998e-06 5.725894e-06
y0   7.279100e-06 4.839527e-05 2.057933e-01 2.550536e-01 2.617471e-01 2.615870e-01
y1   6.058571e-07 4.011714e-06 7.141358e-02 9.646638e-02 9.823928e-02 9.811137e-02
y2   3.774261e-07 2.678236e-06 5.689871e-02 7.605640e-02 8.092440e-02 8.080692e-02
y3   3.631303e-07 2.542625e-06 5.777923e-02 7.711905e-02 7.836441e-02 8.191369e-02
> pval_NLLS2
             yq           y0           y1           y2           y3
yq   7.911587e-26 3.252011e-21 3.593136e-22 2.459477e-22 2.406305e-22
y0            NaN 1.063908e-15 8.239404e-17 5.314785e-17 5.164631e-17
y1            NaN          NaN 6.773484e-18 4.367434e-18 4.309583e-18
y2            NaN          NaN          NaN 2.852917e-18 2.785258e-18
y3            NaN          NaN          NaN          NaN 2.719592e-18
> lambda_NLLS
          1             xq             x0             x1             x2             x3
yq   7.526884e-10 8.555323e-17 1.187795e-15 1.120685e-17 1.110545e-19 1.110255e-21
y0   4.011342e-10 4.578933e-17 6.725100e-16 6.371677e-18 6.319210e-20 6.316897e-22
y1   2.336419e-12 2.665098e-19 3.871296e-18 3.663500e-20 3.634063e-22 3.632761e-24
y2   1.199989e-14 1.367518e-21 1.984097e-20 1.878156e-22 1.861510e-24 1.860846e-26
y3   1.201816e-16 1.369805e-23 1.985610e-22 1.879607e-24 1.864579e-26 1.862287e-28
> lambda_NLLS2
             yq           y0           y1           y2           y3
yq   9.449978e-16 4.884698e-16 2.871187e-18 1.476247e-20 1.478382e-22
y0            NaN 2.518242e-16 1.479292e-18 7.605072e-21 7.616360e-23
y1            NaN          NaN 8.692428e-21 4.469559e-23 4.475442e-25
y2            NaN          NaN          NaN 2.297922e-25 2.301233e-27
y3            NaN          NaN          NaN          NaN 2.304544e-29
```

Sachs et al. Supplementary Materials. S1. Robustness relative to changes in the stem cell proliferation parameter

Parameters used: those of Table 1 exactly as in the main text, so that L_CML=11.8 years.

Representative best fits: red = (y0, x2), green = (y0, y0), blue = (y0, 1)

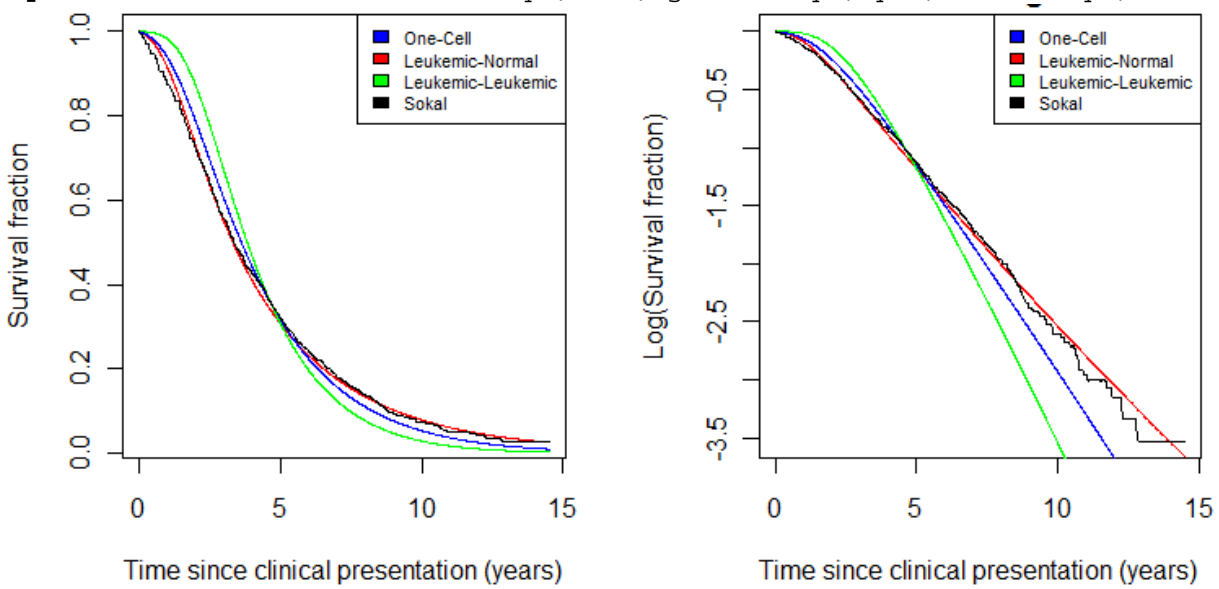





```
> pval_NLLS
          1             xq             x0             x1             x2             x3
yq   5.663263e-12 4.054121e-11 1.359244e-06 4.711422e-06 5.566969e-06 5.706628e-06
y0   2.544350e-03 1.363600e-02 7.407381e-01 8.877695e-01 9.231776e-01 9.228192e-01
y1   3.038664e-04 1.934693e-03 3.983316e-01 5.373731e-01 5.581723e-01 5.575542e-01
y2   1.979984e-04 1.301450e-03 3.393103e-01 4.703171e-01 4.880863e-01 4.874535e-01
y3   1.970655e-04 1.284156e-03 3.427711e-01 4.741323e-01 4.919213e-01 4.912835e-01
> pval_NLLS2
             yq           y0           y1           y2           y3
yq   2.112992e-23 2.267533e-16 2.159905e-17 1.440692e-17 1.415748e-17
y0            NaN 5.588038e-08 4.653020e-09 2.963313e-09 2.963893e-09
y1            NaN          NaN 3.695801e-10 2.373463e-10 2.334569e-10
y2            NaN          NaN          NaN 1.545793e-10 1.486917e-10
y3            NaN          NaN          NaN          NaN 1.465369e-10
> lambda_NLLS
          1             xq             x0             x1             x2             x3
yq   5.184256e-10 5.913423e-17 9.861620e-16 9.122907e-18 9.015779e-20 9.008730e-22
y0   2.445510e-10 2.819071e-17 4.965014e-16 4.648647e-18 4.598718e-20 4.596697e-22
y1   1.415146e-12 1.628537e-19 2.867356e-18 2.680189e-20 2.652622e-22 2.651507e-24
y2   7.263224e-15 8.355223e-22 1.471269e-20 1.374796e-22 1.360776e-24 1.360208e-26
y3   7.271223e-17 8.365338e-24 1.472490e-22 1.375972e-24 1.361944e-26 1.361377e-28
> lambda_NLLS2
             yq           y0           y1           y2           y3
yq   3.872835e-16 1.739092e-16 1.010392e-18 5.185987e-21 5.192671e-23
y0            NaN 7.968001e-17 4.593731e-19 2.355348e-21 2.357635e-23
y1            NaN          NaN 2.651774e-21 1.359667e-23 1.361296e-25
y2            NaN          NaN          NaN 6.970394e-26 6.980818e-28
y3            NaN          NaN          NaN          NaN 6.989011e-30
```

Parameters used: those of Table 1 except that r'=0.015, which implies L_CML=5.35 years.

Representative best fits: red = (y2, x0), green = (y0, y0), blue = (y0,x1).

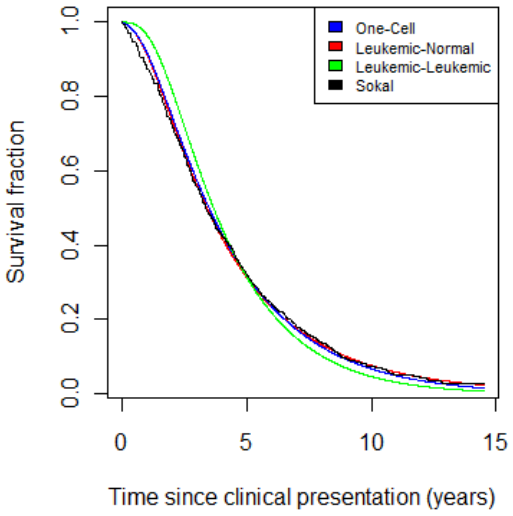


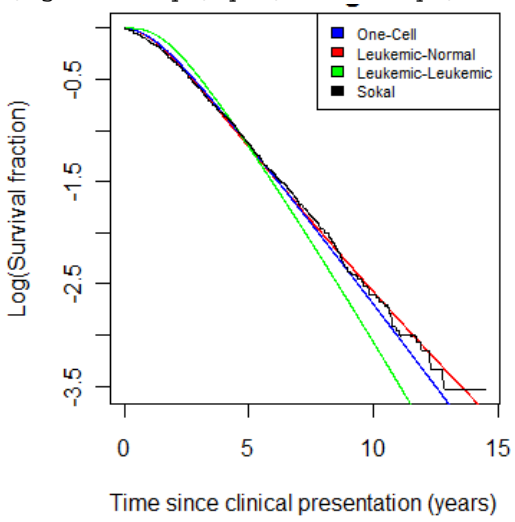

```
> pval_NLLS
          1             xq             x0             x1             x2             x3
yq   2.844264e-08 2.225487e-07 2.187058e-05 0.0000951579 0.0001157081 0.0001186028
y0   3.501038e-01 6.864212e-01 7.694582e-01 0.8214541997 0.8324657651 0.8330325894
y1   1.895591e-01 4.485233e-01 8.433533e-01 0.8016224030 0.7990869706 0.7989853717
y2   1.682193e-01 4.112868e-01 8.910028e-01 0.8318767981 0.8268194491 0.8265935622
y3   1.644237e-01 4.156205e-01 8.937146e-01 0.8336984147 0.8285112987 0.8282790153
> pval_NLLS2
             yq           y0           y1           y2           y3
yq   1.605801e-16 3.943840e-09 2.074408e-09 1.909732e-09 1.899030e-09
y0            NaN 5.575015e-02 2.918580e-02 2.716065e-02 2.659914e-02
y1            NaN          NaN 1.601688e-02 1.450514e-02 1.477847e-02
y2            NaN          NaN          NaN 1.315694e-02 1.340407e-02
y3            NaN          NaN          NaN          NaN 1.314084e-02
> lambda_NLLS
          1             xq             x0             x1             x2             x3
yq   2.516034e-10 2.908031e-17 6.413082e-16 5.876059e-18 5.798879e-20 5.793950e-22
y0   1.359083e-10 1.595933e-17 3.463720e-16 3.205608e-18 3.166946e-20 3.165034e-22
y1   7.581827e-13 8.900385e-20 1.962872e-18 1.828491e-20 1.808088e-22 1.807078e-24
y2   3.863824e-15 4.535484e-22 1.001281e-20 9.338695e-23 9.236097e-25 9.231014e-27
y3   3.869014e-17 4.537901e-24 1.002284e-22 9.348631e-25 9.246005e-27 9.240921e-29
> lambda_NLLS2
             yq           y0           y1           y2           y3
yq   8.071897e-17 4.408330e-17 2.369357e-19 1.199132e-21 1.199890e-23
y0            NaN 2.512569e-17 1.364514e-19 6.915064e-22 6.922041e-24
y1            NaN          NaN 7.365316e-22 3.731226e-24 3.732063e-26
y2            NaN          NaN          NaN 1.889875e-26 1.890289e-28
y3            NaN          NaN          NaN          NaN 1.892042e-30
```

Sachs et al. Supplementary Materials. S1. Robustness relative to changes in the stem cell proliferation parameter

Parameters used: those of Table 1 except that r'=0.03, which implies L_CML=3.19 years.

Representative best fits: red = (y0, xq), green = (y0, y0), blue = (y0, 1).






```
> pval_NLLS
         1            xq           x0           x1           x2           x3
yq   5.162936e-07 3.977753e-06 0.0001119786 0.0004041156 0.0004952833 0.0005094509
y0   6.705753e-01 9.495133e-01 0.9765799992 0.7938912375 0.7638944699 0.7623952991
y1   4.775551e-01 8.394821e-01 0.7599338657 0.8124959501 0.8253116483 0.8259789437
y2   4.395287e-01 8.018277e-01 0.7599376477 0.7890808592 0.7984198170 0.7989151142
y3   4.435909e-01 8.062306e-01 0.7603022728 0.7882631891 0.7974265519 0.7979131593
> pval_NLLS2
             yq           y0           y1           y2           y3
yq   1.253660e-14 1.478062e-07 1.113150e-07 1.081734e-07 1.080590e-07
y0            NaN 2.652946e-01 1.917286e-01 1.777542e-01 1.797365e-01
y1            NaN          NaN 1.393656e-01 1.288145e-01 1.302404e-01
y2            NaN          NaN          NaN 1.233782e-01 1.247315e-01
y3            NaN          NaN          NaN          NaN 1.215572e-01
> lambda_NLLS
         1            xq           x0           x1           x2           x3
yq   1.913928e-10 2.225333e-17 5.275800e-16 4.843551e-18 4.778970e-20 4.774623e-22
y0   1.133036e-10 1.342230e-17 2.994691e-16 2.734700e-18 2.696292e-20 2.694396e-22
y1   6.249185e-13 7.378814e-20 1.723465e-18 1.587664e-20 1.567225e-22 1.566214e-24
y2   3.179625e-15 3.753400e-22 8.812392e-21 8.131556e-23 8.028732e-25 8.023645e-27
y3   3.181197e-17 3.755523e-24 8.822251e-23 8.141347e-25 8.038495e-27 8.033406e-29
> lambda_NLLS2
             yq           y0           y1           y2           y3
yq   4.736768e-17 2.937302e-17 1.545264e-19 7.792261e-22 7.795560e-24
y0            NaN 1.848992e-17 9.921567e-20 5.023644e-22 5.025070e-24
y1            NaN          NaN 5.285242e-22 2.672972e-24 2.673606e-26
y2            NaN          NaN          NaN 1.350463e-26 1.350773e-28
y3            NaN          NaN          NaN          NaN 1.352200e-30
```

The above tables and figures show the following:

- Interaction between leukemic and normal cells remains the preferred mechanism In all cases.
- For r' substantially smaller than the value used in the main calculation, CML latency times are large, but the predictions are not consistent with the data.
- For r' substantially larger (small CML latency times) results are consistent with the data but our approach fails to distinguish sharply among the various alternative blast crisis origination methods.

Next S2. Or jump to:

S2. Stochastic effects

The basic CML/imatinib model [Foo et al. 2009], to which we appended a stochastic blast crisis model, is deterministic, with calculated cell numbers interpreted as expected values. We investigated if this deterministic approach to the cell numbers (subsequently used to calculate blast crisis origination stochastically) was adequate, especially for calculations relevant to early times $\approx$-*L_CML,* where *L_CML* is the CML latency time as defined in Figure 2 of the main text. At these early times one is analyzing a small leukemic clone which starts with a single Ph+ stem cell that has a net growth advantage. Such clones can, early in their development, become extinct due to stochastic small number fluctuations even in the absence of inhibitory density effects and despite their growth advantage (reviewed, e.g., in [Fakir et al. 2009; Lenaerts et al. 2009]). Alternately, they can grow to moderate size, at which point the probability of stochastic extinction becomes negligible. We anticipated that even for these early times a deterministic treatment would be applicable provided we focused on Ph+ clones that grow so large the possibility of stochastic extinction has become negligible. Overall we anticipated that the deterministic treatment of the base model would be applicable to our case for the following reasons: the key data modeled refers to times near the clinical presentation time, when all cell populations involved are large; the initiation of a Ph+ cell is usually far too early to be directly observed; initiation timing is relevant only to the extent that estimating *L_CML* by using radiobiology data can be used to help determine pH+ stem cell growth rates at times near the clinical presentation time; and, as discussed in Section S1, the radiobiology data that determines *L_CML* involves uncertainties substantially larger than a few months.

To test our above anticipations we used a formalism with both deterministic and stochastic variants, and compared the results. Kim et al. [Kim et al. 2008], reformulated a stochastic, agent-based, cell population dynamical CML/imatinib model of Roeder et al. [Roeder et al. 2006]. In the reformulation, they used a stochastic formalism; they compared results to the deterministic limit. They kindly made their MATLAB source code available to us for our investigations.

Distributional Approx*i*mations

Because run-time was an issue [Roeder et al. 2006; Kim et al. 2008] we first increased the stochastic simulation speed ~20-fold by approximating binomial distributions with Poisson or normal distributions respectively when standard conditions (p. 129 or pp. 162-63 respectively in[Devore 1995]) for the applicability of these approximations hold. We tested our modifications, and the stochastic simulation results remain the same, within the fluctuations discussed by Kim et al., after such approximations. After matching the standard test case of one leukemic stem cell among 10^5 normal stem cells, we extended our simulations to the case, believed to be more realistic but previously very demanding on CPU time [Roeder et al. 2006; Kim et al. 2008], of one leukemic stem cell among 10^6 normal stem cells. This case did not require noticeably extra CPU time in our stochastic calculations. Currently, for a typical stochastic simulation, it takes less than 60 mins with a low-end PC for one run.

Probability of Accidental Extinction of a Small Ph+ Clone

A further calculation obtained an analytic estimate of the probability for accidental extinction of a small leukemic clone in the model of [Kim et al. 2008]. Focussing on stem cells during the early steps and neglecting time delays short compared to the times (~ years) of interest to us, the relevant process can be approximated as follows (Figure S1.1).

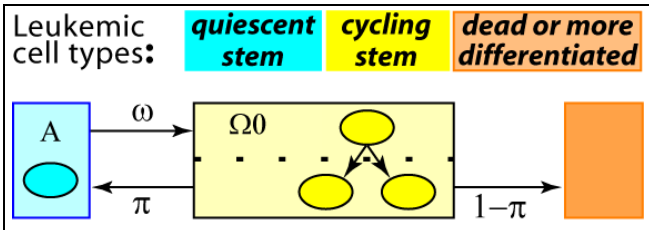


Figure S1.1. Cell dynamics relevant to extinction of small leukemic clones. Here, in the terminology of the references: compartment A is the alpha cell compartment; Ω0 is the omega compartment for the first 2 days, during which a cell that has entered from A divides exactly once; π and 1-π are probabilities a daughter cell in Ω0 undergoes the indicated transition; and the probability of transitions from the orange compartment to the left (i.e. of de-differentiation) is negligible. Ω0 corresponds, at least roughly, to the cycling stem cell compartment of the main text.

From Figure S1.1, we can derive a simple formula to estimate the probability of accidental extinction of a small number of Ph+ stem cells. In the model the starting point is one leukemic stem cell in compartment A. It is seen from Figure S1.1 that whenever a cell leaves A, its clone after 2 more days has no cells in Ω0 and has respectively 0, 1, or 2 cells in A, with respective probabilities $(1-\pi)^2$, $2\pi(1-\pi)$, or π^2. Thus for m cells in A, regarded as wholly independent at these early times where there are as yet no substantial extra density inhibitions due to the leukemic clone, one has a special case of the gambler's ruin, with the chance of extinction being $[(1-\pi)/\pi]^{2m}$. For the parameters used in [Kim et al. 2008] we estimate $\pi \approx 0.56$ corresponding to an extinction probability of ≈ 0.62 when starting with one cell in A and a wholly negligible extinction probability when and if a Ph+ clone reaches 20 cells in A.

The time to extinction starting with one cell can be estimated by simulations, which for the approximations used in Figure S1.1 are very fast. We used the value $\omega \approx 1.46$ per month, based on the parameters in [Kim et al. 2008]. In 10,000 runs we found for the extinction times in months: minimum=0.2, maximum=55, median= 2.0, mean=3.7, and sqrt(variance)=4.7. The published simulations, using the model directly, likewise give average extinction times of a few months. Moreover, CML is a very rare event, even for the Japanese atomic bomb survivors whose excess CML risk data leads to the estimate of *L_CML* (see main text), so the chance of one individual having two CML clones is negligible.

Overall, these estimates showed that in the model of [Kim et al. 2008] the following hold: a) with *L_CML* having order of magnitude of a decade, one can simply interpret *–L_CML* as the time one Ph+ clone reaches 20 cells; and b), from that time point on a deterministic treatment is appropriate, without innacuracies greater than those inherent in the data and data processing.

Stochastic Extinction of Malignant Blast Clones

Similar considerations apply to the blast crisis latency time *L_BC*. When a malignant blast is formed, its clone may become accidentally extinct despite the large relative fitness (growth advantage) of such blasts. In this case the growth advantage is so large that definitive escape from extinction occurs for small clones and occurs rapidly [Fakir et al. 2009]. Thus regarding the critical event as the occurrence of a malignant blast clone so large it has definitively escaped stochastic extinction and regarding the blast crisis latency time as the relevant adjustable parameter, as was done in the main text, is appropriate

Next S#. Or jump to:
S1. Robustness relative to changes in the stem cell proliferation parameter
S2. Stochastic effects
S3. Statistical methods.
S4. Response of main calculation to changes in the blast crisis latency time *L_BC*
S5. Response of main calculation to changes in stem cell transition parameters
S6. Customized computer program used
S7. Supplement bibliography

.

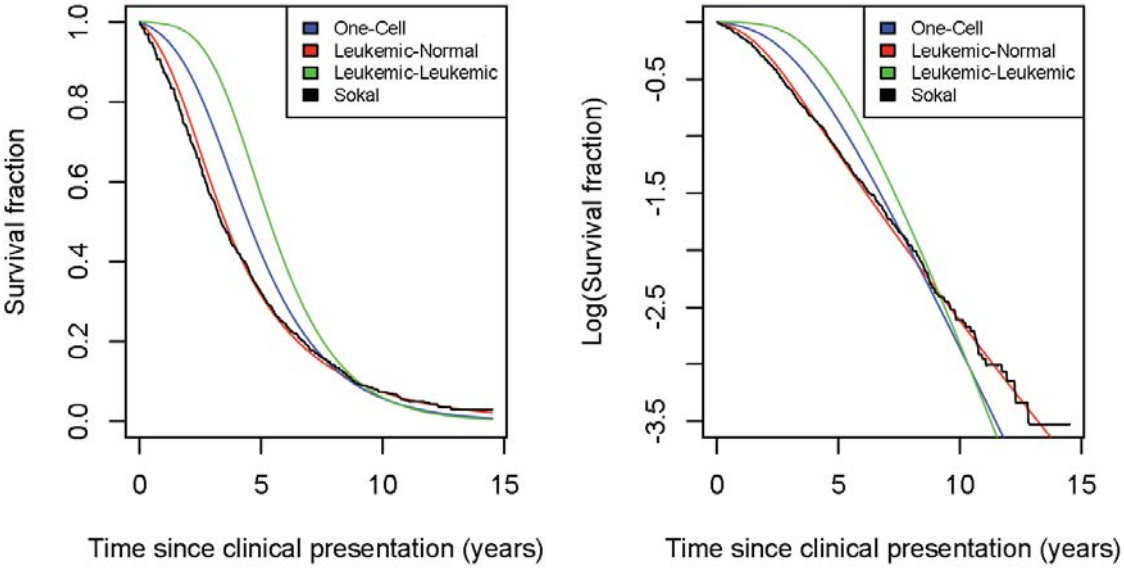
S3. Statistical methods.

Customized R-software blest (blast crisis estimator) calculated first cell numbers and then blast integrals. The latter were used as follows to estimate values of the constant λ for each of the 45 unique mechanism pairs and for various values of *L_BC* from 0 to 5.75 years. For each mechanism, *S* calculated by using blast integrals as described above is matched to the 1580 equispaced time steps over a course of 14.5 years in Sokal's data, using the trapezoidal method. The λ for each mechanism is estimated based on curve fitting to Sokal's automated empirical data by a NLLS [nonlinear least squares] method [Berkson & Gage 1952].To characterize goodness of fit, we applied a log-rank test [Liefers et al. 1998; Orlic et al. 2001], assuming that the theoretical data consist of 1635 patients. The null hypothesis was that at each time step, the fraction of deaths is distributed equally in both curves; the significance level was chosen as p>0.05, and results are shown in Tables 2 and 3 of the main text as well as many tables in S1, S4 and S5.

Additionally, because the data was available only in the form of a semi-logarithmic plot and λ occurs linearly in logS (see above) we applied variance-weighted linear regression (VWLR) to logS for the parameters of Table 1, as a check on the robustness of conclusions drawn from the NLLS approach. Linear regression is a useful model for survival analysis [Buckley & James 1979; Ma 2007], and applies in our analysis after logarithmic transformation to match the Sokal curve.

In applying inverse-variance weights to the logarithm of the data, or $\log\left(\hat{S}(t)\right) = \log\left(\prod_{t_i<t} \frac{n(t_i)-d(t_i)}{n(t_i)}\right)$, we estimated the variance by Greenwood's formula, $\mathrm{var}(t) = \frac{\hat{S}^2(t)[1-\hat{S}(t)]}{n(t_i)}$, where $n(t_i)$ and $d(t_i)$ represent the number of people at risk and dead at time t_i ,respectively. Inverse-variance weighting improves fits because as variability increases, precision decreases, so the corresponding data points should carry less weight in parameter estimation [Almeida et al. 2002]. The variances were tested to be heteroscedastic based on results from the *F*-test ($F_{1580,45,}$ = 26080.46, *p*-value < 0.0001) [Cacoullos 2001]. p-values were calculated as before.

Representative best fits



```
> pval_VWLR
         1             xq           x0            x1            x2           x3
yq   1.758201e-35 3.623846e-29 5.095808e-14 1.383016e-12 2.247649e-12 2.302179e-12
y0   2.799422e-04 3.445117e-03 7.935448e-01 8.511167e-01 8.582099e-01 8.585523e-01
y1   4.869534e-06 1.414653e-04 4.095669e-01 4.855582e-01 4.962451e-01 4.967701e-01
y2   2.168631e-06 7.357216e-05 3.458543e-01 4.202316e-01 4.309180e-01 4.314444e-01
y3   2.079676e-06 7.111338e-05 3.427161e-01 4.169665e-01 4.276464e-01 4.281725e-01
> pval_VWLR2
             yq           y0           y1          y2          y3
yq   4.175915e-112 6.646469e-58 3.153940e-61 8.685702e-62 8.139604e-62
y0             NaN 7.352650e-18 1.515755e-20 5.063522e-21 4.789215e-21
y1             NaN          NaN 2.871364e-23 9.537056e-24 9.018485e-24
y2             NaN          NaN          NaN 3.169068e-24 2.996857e-24
y3             NaN          NaN          NaN          NaN 2.834011e-24
> lambda_VWLR
         1             xq           x0            x1            x2           x3
yq   1.416918e-10 1.863192e-17 5.068100e-16 4.921136e-18 4.897694e-20 4.896525e-22
y0   1.805074e-10 2.228878e-17 4.859130e-16 4.594509e-18 4.554168e-20 4.552167e-22
y1   9.537327e-13 1.185431e-19 2.690352e-18 2.550652e-20 2.529233e-22 2.528170e-24
y2   4.810730e-15 5.986005e-22 1.367614e-20 1.297239e-22 1.286438e-24 1.285902e-26
y3   4.812840e-17 5.988962e-24 1.368749e-22 1.298348e-24 1.287542e-26 1.287006e-28
> lambda_VWLR2
             yq           y0           y1          y2          y3
yq   1.168937e-17 2.238843e-17 1.131609e-19 5.667866e-22 5.668356e-24
y0             NaN 3.349613e-17 1.724519e-19 8.663276e-22 8.665314e-24
y1             NaN          NaN 8.857657e-22 4.448061e-24 4.449024e-26
y2             NaN          NaN          NaN 2.233556e-26 2.234033e-28
y3             NaN          NaN          NaN          NaN 2.234510e-30
```

As is seen by comparing the Figures and Tables with the corresponding NLLS results, both methods gave quite similar results. The trade-off between them is that the nonlinear method of least squares gives better fits to *S* than to log*S* and vice-versa for the variance-weighted linear regression. Our main conclusion, about the preference for leukemic-normal interactions as the mechanism for blast crisis origination remains the same

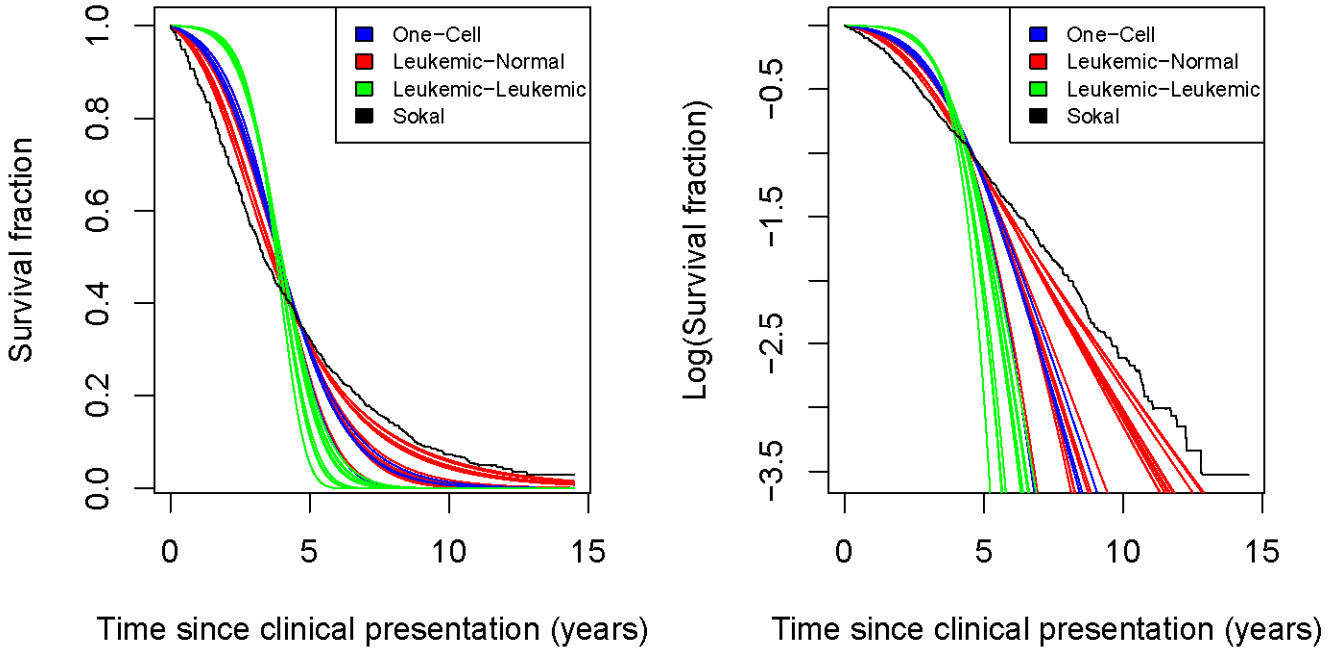
Next S4. Or jump to:

S1. Robustness relative to changes in the stem cell proliferation parameter
S2. Stochastic effects
S3. Statistical methods.
S4. Response of main calculation to changes in the blast crisis latency time *L_BC*
S5. Response of main calculation to changes in stem cell transition parameters
S6. Customized computer program used
S7. Supplement bibliography

S4. Response of main calculation to changes in the blast crisis latency time *L_BC*

Parameters used: Table 1 of the main text except for various non-zero values of the blast crisis latency time L_BC, as specified in each case below. Tables below give details on Table 3 of the main text. Figures are similar to Figure 4AB of the main text.

Blast crisis latency time parameter L_BC = 4 months.



>

pval_NLLS

	1	xq	x0	x1	x2	x3
yq	2.188833e-13	1.384660e-12	2.281388e-07	8.084066e-07	9.712026e-07	9.909190e-07
yq	2.913530e-04	1.793690e-03	5.170430e-01	6.639790e-01	6.746241e-01	6.742425e-01
yq	1.865980e-05	1.330885e-04	2.056170e-01	3.106732e-01	3.228969e-01	3.224011e-01
yq	1.102196e-05	8.381746e-05	1.653894e-01	2.565753e-01	2.759365e-01	2.754497e-01
yq	1.063583e-05	7.988784e-05	1.614115e-01	2.505338e-01	2.694910e-01	2.690097e-01

> pval2_NLLS

	yq	y0	y1	y2	y3
yq	2.597798e-25	1.850962e-18	1.210771e-19	7.564935e-20	7.377383e-20
yq	NaN	5.152968e-10	2.516786e-11	1.461459e-11	1.435006e-11
yq	NaN	NaN	1.158477e-12	6.816011e-13	6.540113e-13
yq	NaN	NaN	NaN	3.941003e-13	3.861666e-13
yq	NaN	NaN	NaN	NaN	3.784398e-13

> lambda_NLLS

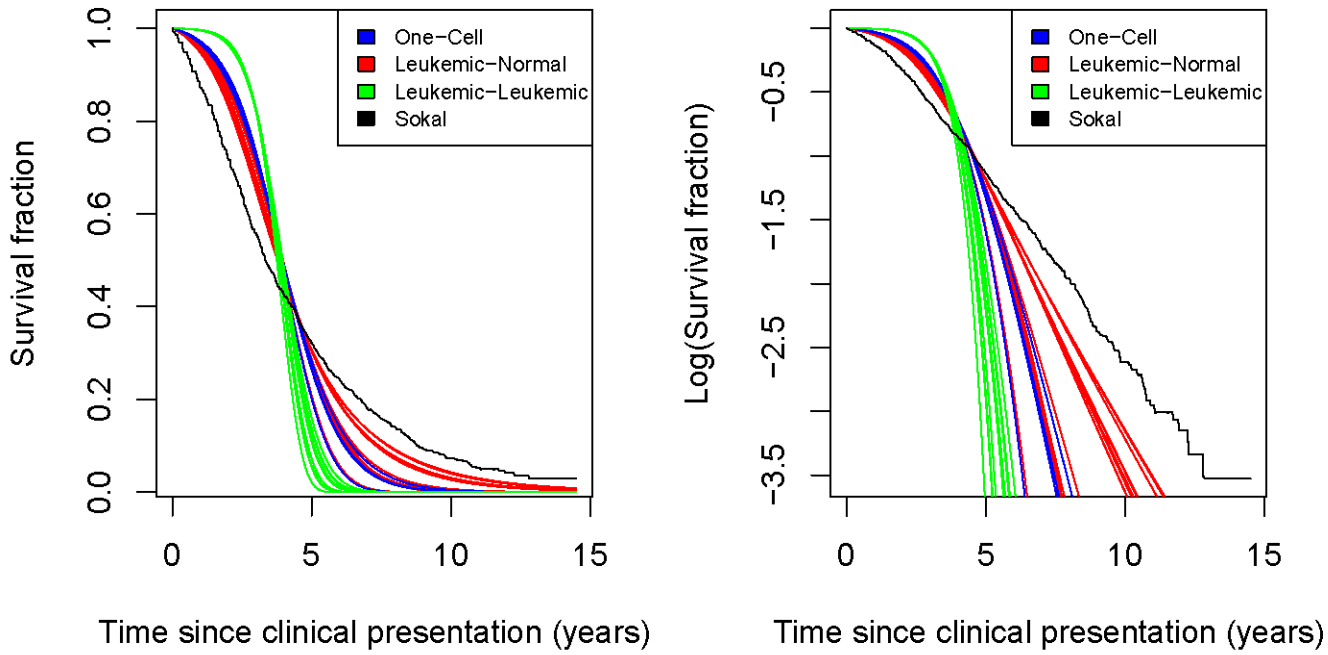
	1	xq	x0	x1	x2	x3
yq	6.570017e-10	7.451575e-17	1.155646e-15	1.067516e-17	1.054588e-19	1.053814e-21
yq	2.833505e-10	3.242676e-17	5.398817e-16	5.049162e-18	5.000399e-20	4.998221e-22
yq	1.671555e-12	1.908665e-19	3.159409e-18	2.947982e-20	2.918049e-22	2.916830e-24
yq	8.608452e-15	9.820134e-22	1.624776e-20	1.515681e-22	1.499066e-24	1.498445e-26
yq	8.622346e-17	9.837558e-24	1.627566e-22	1.518343e-24	1.501708e-26	1.501086e-28

> lambda_NLLS2

	yq	y0	y1	y2	y3
yq	6.160093e-16	2.532203e-16	1.506460e-18	7.765459e-21	7.777620e-23
yq	NaN	1.050702e-16	6.205952e-19	3.196005e-21	3.200404e-23
yq	NaN	NaN	3.674146e-21	1.892330e-23	1.895549e-25
yq	NaN	NaN	NaN	9.748999e-26	9.762837e-28
yq	NaN	NaN	NaN	NaN	9.776679e-30

Sachs et al. Supplementary Materials. S4. Response of main calculation to changes in the blast crisis latency time

blast crisis latency time parameter L_BC = 11 months.




```
          1             xq             x0             x1             x2             x3
yq   3.664402e-16 1.878079e-15 5.889966e-09 1.765555e-08 2.080704e-08 2.105919e-08
yq   1.372511e-06 1.010279e-05 1.595387e-01 2.368287e-01 2.495124e-01 2.492791e-01
yq   2.439859e-08 1.919522e-07 2.770864e-02 5.121187e-02 5.530975e-02 5.517068e-02
yq   1.153257e-08 8.933879e-08 1.894164e-02 3.543460e-02 3.980697e-02 3.969230e-02
yq   1.119939e-08 8.524965e-08 1.865533e-02 3.482950e-02 3.911865e-02 3.900519e-02
> pval2_NLLS
             yq            y0            y1            y2            y3
yq   6.304331e-29 9.446755e-23 3.543426e-24 1.995877e-24 1.941734e-24
yq            NaN 8.687298e-15 1.624088e-16 8.065806e-17 7.771368e-17
yq            NaN           NaN 3.265268e-18 1.631278e-18 1.581452e-18
yq            NaN           NaN           NaN 8.242306e-19 7.955870e-19
yq            NaN           NaN           NaN           NaN 7.681295e-19
> lambda_NLLS
          1             xq             x0             x1             x2             x3
yq   1.039863e-09 1.168999e-16 1.576152e-15 1.457053e-17 1.439499e-19 1.438568e-21
yq   3.853742e-10 4.352254e-17 6.442000e-16 6.016498e-18 5.953876e-20 5.951298e-22
yq   2.367019e-12 2.666985e-19 3.871926e-18 3.604992e-20 3.566233e-22 3.564748e-24
yq   1.228199e-14 1.383667e-21 2.001136e-20 1.863089e-22 1.841762e-24 1.841002e-26
yq   1.230395e-16 1.386404e-23 2.004361e-22 1.866163e-24 1.844812e-26 1.844051e-28
> lambda_NLLS2
             [,1]          [,2]          [,3]          [,4]          [,5]
yq   1.530147e-15 5.464930e-16 3.404654e-18 1.770120e-20 1.773601e-22
yq            NaN 1.938299e-16 1.204267e-18 6.258285e-21 6.270736e-23
yq            NaN           NaN 7.495000e-21 3.896536e-23 3.904045e-25
yq            NaN           NaN           NaN 2.025601e-25 2.029639e-27
yq            NaN           NaN           NaN           NaN 2.033677e-29
```

blast crisis latency time parameter L_BC = 19 months.

```
> pval_NLLS
         1            xq           x0           x1           x2           x3
yq   1.101435e-19 4.103628e-19 2.072249e-11 3.096604e-11 3.225774e-11 3.241649e-11
y0   1.243812e-10 7.686782e-10 7.080828e-03 1.183519e-02 1.255602e-02 1.253816e-02
y1   5.812938e-13 3.461015e-12 2.523101e-04 4.815518e-04 5.139200e-04 5.120515e-04
y2   2.164387e-13 1.276231e-12 1.302096e-04 2.428558e-04 2.688868e-04 2.677853e-04
y3   2.091377e-13 1.204672e-12 1.258046e-04 2.337716e-04 2.586964e-04 2.576301e-04
> pval2_NLLS
             yq           y0           y1           y2           y3
yq   3.490554e-33 2.077174e-28 7.476664e-30 4.132180e-30 4.008359e-30
y0            NaN 3.533215e-22 5.059009e-24 2.389979e-24 2.293519e-24
y1            NaN          NaN 9.167114e-26 4.503241e-26 4.336160e-26
y2            NaN          NaN          NaN 2.219968e-26 2.146190e-26
y3            NaN          NaN          NaN          NaN 2.069146e-26
> lambda_NLLS
         1            xq           x0           x1           x2           x3
yq   1.888437e-09 2.108157e-16 2.404379e-15 2.244525e-17 2.221337e-19 2.220118e-21
y0   6.067739e-10 6.776406e-17 8.392969e-16 7.840638e-18 7.760821e-20 7.757448e-22
y1   3.928972e-12 4.381462e-19 5.255443e-18 4.898721e-20 4.848023e-22 4.845994e-24
y2   2.059014e-14 2.295655e-21 2.737206e-20 2.551926e-22 2.524147e-24 2.523098e-26
y3   2.063592e-16 2.301329e-23 2.742936e-22 2.557382e-24 2.529559e-26 2.528509e-28
> lambda_NLLS2
             [,1]         [,2]         [,3]         [,4]         [,5]
yq   5.066854e-15 1.607107e-15 1.052520e-17 5.524365e-20 5.538095e-22
y0            NaN 5.021598e-16 3.296517e-18 1.730765e-20 1.735164e-22
y1            NaN          NaN 2.164053e-20 1.136249e-22 1.139117e-24
y2            NaN          NaN          NaN 5.966321e-25 5.981050e-27
y3            NaN          NaN          NaN          NaN 5.996071e-29
```

blast crisis latency time parameter L_BC = 29 months.

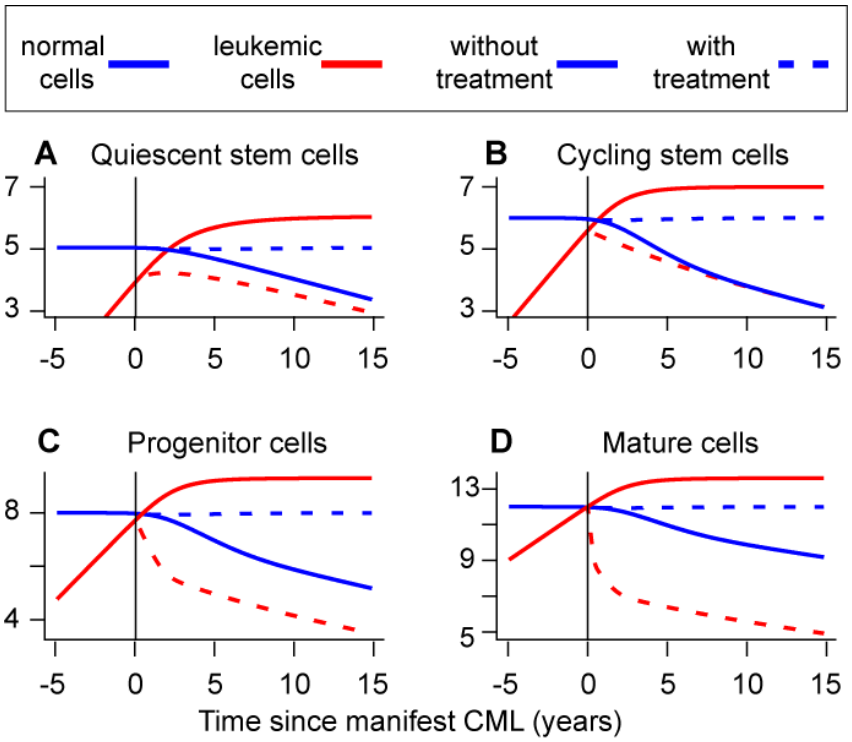
```
> pval_SSE
             [,1]         [,2]         [,3]         [,4]         [,5]         [,6]
[1,] 6.027362e-24 1.407657e-23 6.459697e-16 1.628283e-16 1.314322e-16 1.299599e-16
[2,] 2.122350e-17 7.122967e-17 2.847740e-07 1.596464e-07 1.434487e-07 1.435789e-07
[3,] 1.155951e-19 3.476919e-19 8.471658e-10 3.433297e-10 2.868515e-10 2.868649e-10
[4,] 4.542704e-20 1.332220e-19 2.775850e-10 1.046993e-10 8.770528e-11 8.769680e-11
[5,] 4.313882e-20 1.266979e-19 2.603557e-10 9.820877e-11 8.517503e-11 8.227165e-11
> pval2_SSE
             [,1]         [,2]         [,3]         [,4]         [,5]
[1,] 1.245502e-37 6.005593e-35 6.010816e-36 3.941214e-36 3.856245e-36
[2,]          NaN 1.156578e-31 6.944480e-33 4.159719e-33 4.052155e-33
[3,]          NaN          NaN 5.026084e-34 3.110865e-34 3.035798e-34
[4,]          NaN          NaN          NaN 1.937949e-34 1.890934e-34
[5,]          NaN          NaN          NaN          NaN 1.845854e-34
```

Next S5. Or jump to:

S1. Robustness relative to changes in the stem cell proliferation parameter
S2. Stochastic effects
S3. Statistical methods.
S4. Response of main calculation to changes in the blast crisis latency time *L_BC*
S5. Response of main calculation to changes in stem cell transition parameters
S6. Customized computer program used
S7. Supplement bibliography

S5. Response of main calculation to changes in stem cell transition parameters

A main consideration for Foo et al. [Foo et al. 2009] in choosing α and β was that the ratio α/β equals the equilibrium ration of quiescent to cycling stem cells, for both normal and leukemic stem cells, with values α/β >1 preferred on biological grounds. However in their analysis they consider also the possibility α=0.0001 per day and β=0.0009 per day so that α/β =1/9, instead of 9 as above. To test robustness of the conclusions from our main calculation we repeated the analysis with these values, retaining all the other parameter values of our main calculation. We confirmed robustness of the conclusion that leukemic-normal interactions are the preferred blast crisis origination mechanism, as follows. First, a CML mean latency time of *L_CML*= 9.64 years and the cell numbers shown in the figure were found

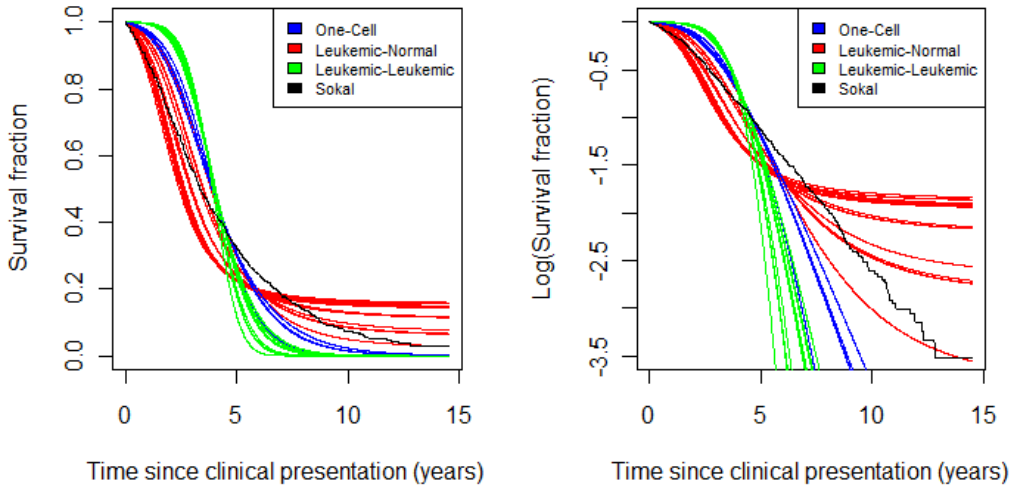


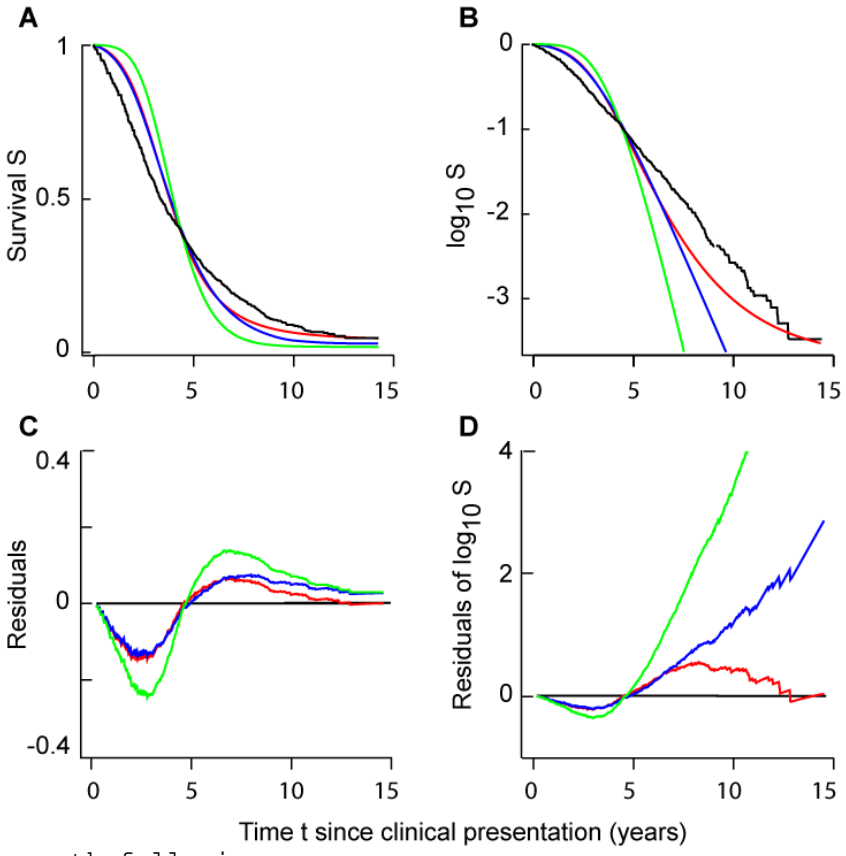
Proceeding just as before we again found *p*-values. The key ones are shown in the table.

	y_q*	y_0	y_1	1	x_q	x_0	x_1
y_q	<E-17	<E-12	<E-13	<E-8	***0.94***	0.32	0.31
y_0		***<E-5***	<E-6	***6.4E-3***	0.34	0.85	0.79
y_1			<E-7	7.7E-4	0.25	0.69	0.63

It is seen that leukemic-normal interactions are again strongly favored. For leukemic-normal interaction mechanisms, the highest *p*-values here occurred with x_q involved. Thus, while the conclusion that pairwise interactions between leukemic and normal cells is the preferred general mechanism remains unaltered, the approach here may not be able to distinguish sharply between more specific alternatives. We again selected 3 preferred representatives of the three main kinds of interactions (boldface in Table). The figure below shows the fitted curves.

Sachs et al. Supplementary Materials. S5. Response of main calculation to changes in stem cell transition parameters





Values for lambda were th following:

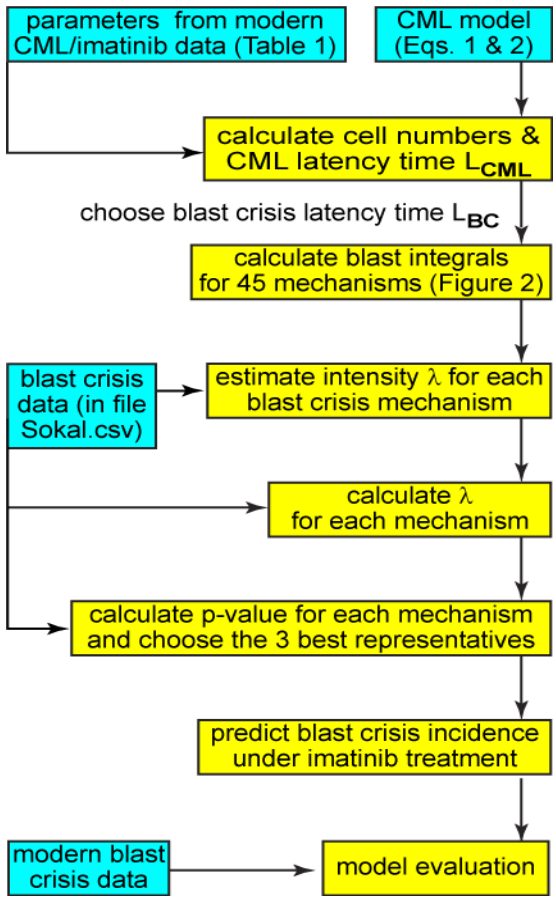
	1	xq	x0	x1	x2	x3
yq	4.103679e-09	5.488754e-14	2.053876e-14	1.694765e-16	1.647555e-18	1.645260e-20
y0	1.533834e-10	2.391485e-15	7.557001e-16	6.326958e-18	6.167668e-20	6.160175e-22
y1	8.927709e-13	1.362066e-17	4.579060e-18	3.811224e-20	3.710380e-22	3.705638e-24
y2	4.586059e-15	6.972407e-20	2.368044e-20	1.969288e-22	1.916711e-24	1.914239e-26
y3	4.591134e-17	6.986925e-22	2.372661e-22	1.972950e-24	1.920252e-26	1.917774e-28

	yq	y0	y1	y2	y3
yq	2.194377e-14	7.944584e-16	4.593447e-18	2.355545e-20	2.358991e-22
y0	NaN	2.952952e-17	1.696581e-19	8.692100e-22	8.701341e-24
y1	NaN	NaN	9.750973e-22	4.996023e-24	5.000778e-26
y2	NaN	NaN	NaN	2.559494e-26	2.562812e-28
y3	NaN	NaN	NaN	NaN	2.565202e-30

Next S6. Or jump to:

S1. Robustness relative to changes in the stem cell proliferation parameter
S2. Stochastic effects
S3. Statistical methods.
S4. Response of main calculation to changes in the blast crisis latency time *L_BC*
S5. Response of main calculation to changes in stem cell transition parameters
S6. Customized computer program used
S7. Supplement bibliography

S6. Customized computer program used



Flow chart.

The flow chart to the left is for the R program used to implement our calculations. Blue shows input; yellow shows calculations. Both boxes in the first line are taken directly from [Foo et al. 2009]. References, e.g. "Table 1" are to items in the main text, which also defines the blast crisis intensity parameter λ.

We suggested in the main text that because in effect our choice of blast crisis mechanism and resulting prediction of the results on blast crisis origination during imatinib treatment involves only one parameter adjustment (namely T_{BC}≡L_BC=0), the fit obtained can count as validation. A caveat is that we have not shown robustness with respect to the choice among various modern CML/imatinib models in the literature.

The program uses the auxiliary file Sokal.csv, estimating survival in an extensive data set from the pre-imatinib era (see main text, Methods section).

```
## Program blest (blast crisis estimator) to analyze CML blast crisis.
## Copyright (C) 2010 Kerstin Johnsson, Janet Luo and Rainer Sachs
## This program is free software; you can redistribute it and/or
## modify it under the terms of the GNU General Public License
## as published by the Free Software Foundation; version 2.

## This program is distributed in the hope that it will be useful,
## but WITHOUT ANY WARRANTY; without even the implied warranty of
## MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
## GNU General Public License http://www.r-project.org/ for more details.
## Contact Prof. R. Sachs, sachs@math.berkeley.edu for questions or comments.

## Implements the paper "Intercellular interaction mechanisms for the
## origination of blast crisis in chronic myeloid leukemia"

## This script requires the auxiliary file Sokal.csv.
```

```
##It simulates the number of Ph- and Ph+ stem, progenitor, differentiated, ##and mature
cells for untreated and imatinib-treated chronic myelogenous ##leukemia (CML) based on a
model of Foo et al.
## (PLoS Comput Biol 2009, 5:e1000503); it also estimates the probability
## of getting blast crisis for various blast production mechanisms.

## The variables used are the following:
##
## t:Number of days after disease initiation; (in the paper there is a different ##time
origin chosen, so that tpaper=tblest +L_CML, where L_CML=ttr is CML latency time.
## x0:   Number of Ph- stem cells (no treatment). In paper x_0, etc.
## x0tr: Number of Ph- stem cells (treatment)
## x1:   Number of Ph- progenitor cells (no treatment)
## x1tr: Number of Ph- progenitor cells (treatment)
## x2:   Number of Ph- differentiated cells (no treatment)
## x2tr: Number of Ph- differentiated cells (treatment)
## x3:   Number of Ph- terminally differentiated cells (no treatment)
## x3tr: Number of Ph- terminally differentiated cells (treatment)
## y0:   Number of Ph+ stem cells (no treatment)
## y0tr: Number of Ph+ stem cells (treatment)
## y1:   Number of Ph+ progenitor cells (no treatment)
## y1tr: Number of Ph+ progenitor cells (treatment)
## y2:   Number of Ph+ differentiated cells (no treatment)
## y2tr: Number of Ph+ differentiated cells (treatment)
## y3:   Number of Ph+ terminally differentiated cells (no treatment)
## y3tr: Number of Ph+ terminally differentiated cells (treatment)
## diagnosislim: number of terminally differentiated leukemic cells required for
diagnosis
## tdtr: time when treatment is discontinued

library(odesolve)
## If this results in an error message, run install.packages('odesolve')

## Program can model discontinuation of treatment after tdtr days.
## (No discontinuation tdtr = Inf)
tdtr = Inf # Time t(from beginning of simulation) at which treatment is discontinued
# tdtr = ttr + 4*365 # Run first with tdtr = Inf to determine ttr

## Parameters (from Foo et al. 2009)
x0eq = 10^6; y0eq = 10^7;
x1eq = 10^8;
x2eq = 10^10;
x3eq = 10^12

diagnosislim = 10^12 # number of terminally differentiated cells required for diagnosis
and treatment

rx = .005; ry = .008; rytr = .002; # ry=r' in the text. If rytr < ry, treatment affects
stem cells
alpha = .0009; beta = .0001
d0 = .003; d1 = .008; d2 = .05; d3 = 1;
##blast crisis latency time L_BC is not in Foo et al. It is given later in this program
ax = d1*x1eq/x0eq
ay = 2*ax; aytr = ay/100
bx = d2*x2eq/x1eq
by = 2*bx; bytr = by/750
cx = d3*x3eq/x2eq
cy = cytr = cx; # Leukemia or treatment does not affect differentiated cells
px = (rx/d0 - 1)/x0eq# makes ODE have x0=x0eq at equilibrium
py = (ry/d0 - 1)/y0eq

## Initial values (at time of CML initiation)
```

```
x0init = x0trinit = x0eq
xqinit = xqtrinit = alpha*x0eq/beta
x1init = x1trinit = x1eq
x2init = x2trinit = x2eq
x3init = x3trinit = x3eq
y0init = y0trinit = 1 # disease initiation - one leukemic stem cell, not quiescent
yqinit = yqtrinit = y1init = y1trinit = y2init = y2trinit = y3init = y3trinit = 0

## Initial value for F = (x0,x0tr,xq,xqtr,x1,x1tr,x2,x2tr,x3,x3tr,
##                        y0,y0tr,yq,yqtr,y1,y1tr,y2,y2tr,y3,y3tr,
##                        )

Finit = c(x0init, x0trinit, xqinit, xqtrinit, x1init, x1trinit, x2init, x2trinit, x3init,
x3trinit, y0init, y0trinit, yqinit, yqtrinit, y1init, y1trinit, y2init, y2trinit, y3init,
y3trinit)

treatm = FALSE # no treatment initially

## dF/dt = f(t,F)
f = function(t,F,par = c(alpha, beta,
                d0, d1, d2, d3,
                rx, ry, rytr,
                px, py,
                ax, ay, aytr,
                bx, by, bytr,
                cx, cy, cytr)) {

x0 = F[1]; x0tr = F[2]
xq = F[3]; xqtr = F[4]
x1 = F[5]; x1tr = F[6]
x2 = F[7]; x2tr = F[8]
x3 = F[9]; x3tr = F[10]

y0 = F[11]; y0tr = F[12]
yq = F[13]; yqtr = F[14]
y1 = F[15]; y1tr = F[16]
y2 = F[17]; y2tr = F[18]
y3 = F[19]; y3tr = F[20]

alpha = par[1]; beta = par[2]
d0 = par[3]; d1 = par[4]; d2 = par[5]; d3 = par[6]
rx = par[7]; ry = par[8]; rytr = par[9]
px = par[10]; py = par[11]
ax = par[12]; ay = par[13]; aytr = par[14]
bx = par[15]; by = par[16]; bytr = par[17]
cx = par[18]; cy = par[19]; cytr = par[20]

if(y3 >= diagnosislim) {
   treatm = TRUE
}

if(!treatm | t >= tdtr) {# before diagnosis and after discontinuation there is no
treatment
   rytr = ry
   aytr = ay
   bytr = by
   cytr = cy
}

dx0 = (rx/(1 + px*(x0 + y0))-d0-alpha)*x0 + beta*xq
dx0tr = (rx/(1 + px*(x0tr + y0tr))-d0-alpha)*x0tr + beta*xqtr
dxq = alpha*x0 - beta*xq
```

```
dxqtr = alpha*x0tr - beta*xqtr
dx1 = ax*x0 - d1*x1
dx1tr = ax*x0tr - d1*x1tr
dx2 = bx*x1 - d2*x2
dx2tr = bx*x1tr - d2*x2tr
dx3 = cx*x2 - d3*x3
dx3tr = cx*x2tr - d3*x3tr

dy0 = (ry/(1 + py*(x0 + y0))-d0-alpha)*y0 + beta*yq
dy0tr = (rytr/(1 + py*(x0tr + y0tr))-d0-alpha)*y0tr + beta*yqtr
dyq = alpha*y0 - beta*yq
dyqtr = alpha*y0tr - beta*yqtr
dy1 = ay*y0 - d1*y1
dy1tr = aytr*y0tr - d1*y1tr
dy2 = by*y1 - d2*y2
dy2tr = bytr*y1tr - d2*y2tr
dy3 = cy*y2 - d3*y3
dy3tr = cytr*y2tr - d3*y3tr

   return(list(c(dx0,dx0tr,dxq,dxqtr,dx1,dx1tr,dx2,dx2tr,dx3,dx3tr,
                 dy0,dy0tr,dyq,dyqtr,dy1,dy1tr,dy2,dy2tr,dy3,dy3tr),treatm))
}

maxtime= 37*365
t = 0:maxtime+6

## Solve the ODE dF/dt = f(t,F) with F(0) = Finit
## The result is a matrix with 22 columns: (t, F). F has 20 columns.
## Each column contains the values of the variable for every time value in t.

res = lsoda(Finit,t,f,parms = c(alpha, beta,
                      d0, d1, d2, d3,
                      rx, ry, rytr,
                      px, py,
                      ax, ay, aytr,
                      bx, by, bytr,
                      cx, cy, cytr))

F = res[,2:21]

treatment = res[,22]
ttr = t[min(which(treatment == 1))] # finds the time at which treatment was inserted

x0 = F[,1]; x0tr = F[,2]
xq = F[,3]; xqtr = F[,4]
x1 = F[,5]; x1tr = F[,6]
x2 = F[,7]; x2tr = F[,8]
x3 = F[,9]; x3tr = F[,10]

y0 = F[,11]; y0tr = F[,12]
yq = F[,13]; yqtr = F[,14]
y1 = F[,15]; y1tr = F[,16]
y2 = F[,17]; y2tr = F[,18]
y3 = F[,19]; y3tr = F[,20]

## ***** Plotting cell numbers *****
lowlim3=ttr-5*365-1

highlim3=ttr+15*365-3

pdf(file = 'cellno.pdf', width =6, height =5)

par(mfrow = c(2,2),bty="l")
```

```
xqp= cbind(xq,xqtr,yq,yqtr)

x0p= cbind(x0,x0tr,y0,y0tr)

x1p= cbind(x1,x1tr,y1,y1tr)

x3p= cbind(x3,x3tr,y3,y3tr)



## quiescent stem cells

matplot(t[lowlim3:highlim3]/365,xqp[lowlim3:highlim3,],log = 'y',type = 'l',lwd = 2,col =
c(1,1,2,2),lty = c(1,2,1,2),ylim=c(10^3,2*10^7),

        ylab = 'Number of cells',xlab = '',main = 'A          Quiescent stem
cells',bty="l")

##legend('bottomright',legend = c('Ph- stem cells', 'Ph+ stem cells','Without treatment',
##'With treatment'),lty = c(1,1,1,2),lwd = 3,col = c(1,2,1,1))

abline(v = c(ttr/365),col = 'black', lty = 1)



## cycling stem cells

matplot(t[lowlim3:highlim3]/365,x0p[lowlim3:highlim3,],log = 'y',type = 'l',lwd = 2,col =
c(1,1,2,2),lty = c(1,2,1,2),ylim=c(10^3,2*10^7),

ylab = '',xlab = '',main = 'B          Cycling stem cells',bty="l")

abline(v = c(ttr/365),col = 'black', lty = 1)



## progenitor cells

matplot(t[lowlim3:highlim3]/365,x1p[lowlim3:highlim3,],log = 'y',type = 'l',lwd = 2,col =
c(1,1,2,2),lty = c(1,2,1,2),

        ylab = 'Number of cells',xlab = 't (years)',main = 'C          Progenitor
cells',bty="l")

abline(v = c(ttr/365),col = 'black', lty = 1)



# terminally differentiated cells

matplot(t[lowlim3:highlim3]/365,x3p[lowlim3:highlim3,],log = 'y',type = 'l',lwd = 2,col =
c(1,1,2,2), lty = c(1,2,1,2),

        ylab = '',xlab = 't (years)',main = 'D          Mature cells',bty="l")

abline(v = c(ttr/365),col = 'black', lty = 1)



dev.off()


##Systematic way to calculate all the blast integrals. Then use them to estimate
probability of onset of
##blast crisis.  leukNum=numbers of leukemic cells of various types at various times.
```

```
norm = normal
##following needs corrections to normalize blast integrals to zero at ttr and to ln(.5)
at the appropriate Sokal
## time for untreated leukemia.
leukNum<-cbind(yq,y0,y1,y2,y3)
normNum<-cbind(1,xq,x0,x1,x2,x3)
leukNumtr<-cbind(yq,y0tr,y1tr,y2tr,y3tr)
normNumtr<-cbind(1,xqtr,x0tr,x1tr,x2tr,x3tr)

## 45 'blast integrals' are first calculated, one for each mechanism.
## The blast integral is defined as
## LAMBDA(t) = int y_a(t) dt if the mechanism is a one cell
## event in a cell in a compartment of size y_a and
## LAMBDA(t) = int_0^t y_a(t)*z_b(t) dt if the mechanism is a
## cell-cell interaction between a cell in a compartment
## of size y_a and a cell in a compartment of size z_b.

## The blast integrals are computed by solving an ODE system:
## LAMBDA'(t) = y_a(t) or LAMBDA'(t) = y_a(t)*y_b(t) with the
## initial value LAMBDA(t) = 0.

L=0  # blast crisis latency time L_BC in days

products<-array(0,dim=c(maxtime-ttr+1,5,6))
for (i in 1:5) {
for (j in 1:6) {
for (k in (ttr:maxtime)-L) {
products[k-ttr+L+1,i,j]<- leukNum[k,i]*normNum[k,j]
}}}

blIntegrals<-array(0,dim=c(maxtime-ttr+1,5,6))
for (i in 1:(maxtime-ttr)) {
blIntegrals[i+1,,]=(products+blIntegrals)[i,,]
}

products2<-array(0,dim=c(maxtime-ttr+1,5,5))
for (i in 1:5) {
for (j in i:5) {
for (k in (ttr:maxtime)-L) {
products2[k-ttr+L+1,i,j]<- leukNum[k,i]*leukNum[k,j]
}}}

blIntegrals2<-array(0,dim=c(maxtime-ttr+1,5,5))
for (i in 1:(maxtime-ttr))  {
blIntegrals2[i+1,,]=(products2+blIntegrals2)[i,,]
}

productstr<-array(0,dim=c(maxtime-ttr+1,5,6))  # Similar products as products, except
with treatment
for (i in 1:5) {
for (j in 1:6) {
for (k in (ttr:maxtime)-L) {
productstr[k-ttr+L+1,i,j]<- leukNumtr[k,i]*normNumtr[k,j]
}}}

blIntegralstr<-array(0,dim=c(maxtime-ttr+1,5,6)) # Similar blast integrals as
blIntegrals, except with treatment
for (i in 1:(maxtime-ttr)) {
blIntegralstr[i+1,,]=(productstr+blIntegralstr)[i,,]
}
```

```
productstr2<-array(0,dim=c(maxtime-ttr+1,5,5))  # Similar products as products2, except
with treatment
for (i in 1:5) {
for (j in i:5) {
for (k in (ttr:maxtime)-L) {
productstr2[k-ttr+L+1,i,j]<- leukNumtr[k,i]*leukNumtr[k,j]
}}}

blIntegralstr2<-array(0,dim=c(maxtime-ttr+1,5,5)) # Similar blast integrals as
blIntegrals2, except with treatment
for (i in 1:(maxtime-ttr))  {
blIntegralstr2[i+1,,]=(productstr2+blIntegralstr2)[i,,]
}

# Fitting to Sokal Data [Sokal 1987]
data = read.csv("Sokal.csv")
x = data$X0         # time, where 109 equi-spaced points correspond to 1 year, for 15
years, so there are 109*15=1635 points in all.

y = data$X2.36975   # survival percentages for each of the points

# y[1:6]=rep(y[7],6)  # To make survival fractions never increase, we set the first 6
values, which are increasing, to equal the 7th value. After the 7th value of 97.3976, the
other survival percentages are non-increasing.
x=x[55:length(x)]   # Sokal 6 months = our t=0 = blest t=ttr. Delete the first 6 months
of Sokal
y=y[55:length(y)]   # Delete also the corresponding first 6 months of survival
percentages
x=x-x[1]  # To let the time since clinical presentation start at 0
y=y/y[1]  # To scale all the survival percentages into fractions so that the first
survival fraction = 1

# Calculate S-hat and var(S-hat)
n = y[1:length(y)]*1635  # Number of people at risk at time i
y1 = y[2:length(y)]  # Shifts the time scale by 1 unit to the right because we want to
calculate the number of people who die at time i
y2 = y[1:(length(y)-1)]
d  = (y2-y1)*1635 # Number of people who die at time i
d = c(0, d)

S_hat = 0   # Kaplan-Meier estimate of survival S
for (i in 1:length(d)) { S_hat[i] = prod(((n-d)/n)[1:i])}
summand = d/(n*(n-d))
summation = 0
for (i in 1:length(d)) { summation[i] = sum(summand[1:i]) }
varKM_Greenwood = (S_hat^2*summation)  # var(S-hat) with Greenwood formula
# varKM_Peto = (S_hat)*(1-S_hat)/n  # var(S-hat) with Peto formula, not used.
varlnS = (1/y[1:1580]^2)*varKM_Greenwood[1:1580]  # var (ln(S-hat)) given var(S-hat)


# In general, var(f(X)) ~= (f'(EX))^2 * var(X)
# f(t) = log(t), so f'(t) is 1/t, and (f'(t))^2 = 1/t^2
# var(X) = var(S-hat)
# EX = y, since the Kaplan-Meier estimate is unbiased.
# Thus, var(ln(S-hat)) = (1/y^2)*(var(S-hat))

varlnS = (1/y[1:1580]^2)*varKM_Greenwood[1:1580]


## Trapezoidal Method to match time steps of blIntegrals with Sokal
# 14.5 years correspond to the first 365*14.5=5292 points in blIntegrals, and
14.5*109=1581 points in Sokal.
```

```
# Want to match a blIntegrals (365 points per year) value to the time steps used in Sokal
(109 points per year).
# The newly matched blIntegrals are saved as blIntegrals3 (for blIntegrals) and
blIntegrals4 (for blIntegrals2).


# blIntegrals[floor(Bjumps),i,j] and blIntegrals[floor(Bjumps)+1,i,j] are sandwiched by
the Sokal value we are seeking to match.
# blIntegrals[floor(Bjumps),i,j] refers to the greatest blIntegral value less than the
Sokal value we are seeking to match.
# blIntegrals[floor(Bjumps)+1,i,j] refers to the least blIntegral value greater than the
Sokal value we are seeking to match.


c=5291/1580   # Slope for 14.5 years of blIntegrals or Sokal
blIntegrals3=array(0, dim=c(1581,5,6))
blIntegrals3[1,,]=blIntegrals[1,,]  # This equals 0 anyway.
for (i in 1:5) { for (j in 1:6) { for (k in 2:1581) {
Bjumps=c*(k-1)  # Number of jumps in Blest for k-1 jumps in Sokal

# First blest+ fraction in x from lower index to upper index * difference in their
corresponding y

# Bjumps-floor(Bjumps) is the fraction in x from lower index to upper index
# (blIntegrals[floor(Bjumps)+1,i,j]-blIntegrals[floor(Bjumps),i,j])    is the difference
in y.

blIntegrals3[k,i,j]=blIntegrals[floor(Bjumps),i,j]+(Bjumps-
floor(Bjumps))*(blIntegrals[floor(Bjumps)+1,i,j]-blIntegrals[floor(Bjumps),i,j])

}}}

# Repeat the same process for blIntegrals2 to make blIntegrals4.

blIntegrals4=array(0, dim=c(1581,5,5))
blIntegrals4[1,,]=blIntegrals2[1,,]
for (i in 1:5) { for (j in i:5) { for (k in 2:1581) {
Bjumps=c*(k-1)
blIntegrals4[k,i,j]=blIntegrals2[floor(Bjumps),i,j]+(Bjumps-floor(Bjumps)) *
(blIntegrals2[floor(Bjumps)+1,i,j]-blIntegrals2[floor(Bjumps),i,j])
}}}
# Estimating lambda with variance weighted least squares using the pre-determined formula
for lambda_weighted
# The results are recorded in matrices lambda_weighted of dim 5x6 and lambda_weighted2 of
dim 5x5.


lambda_weighted = matrix(0, nrow=5, ncol=6)
for (i in 1:5) { for (j in 1:6) {
lambda_weighted[i,j]=-sum((log(y[2:1580]))*(blIntegrals3[2:1580,i,j])/varlnS[2:1580])
/sum((blIntegrals3[2:1580,i,j]^2)/varlnS[2:1580])
}}
lambda_weighted2 = matrix(0, nrow=5, ncol=5)
for (i in 1:5) { for (j in i:5) {
lambda_weighted2[i,j]=-
sum((log(y[2:1580]))*(blIntegrals4[2:1580,i,j])/varlnS[2:1580])/sum((blIntegrals4[2:1580,
i,j]^2)/varlnS[2:1580])
}}


# P-value calculation from log-rank test
# Follows http://www.mas.ncl.ac.uk/~njnsm/medfac/docs/surv.pdf, pages 5-7
```

```
# Turn y, lambda_weighted, and lambda_weighted2 into arrays of dim c(1580, 5, 6) or
c(1580, 5, 5) in order to match the dimensions of the predicted survival fractions.
Within each i,j, we want to create a 2x2 table for each of the 1580 fractions, one for
each time value, like the one on page 5 of surv.pdf.

yarray = array(0, dim=c(1580, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 2:1580) {
yarray[k,i,j] = y[k]
}}}
lambdaarray_weighted = array(0, dim=c(1580, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 2:1580) {
lambdaarray_weighted[k,i,j] = lambda_weighted[i,j]
}}}
lambdaarray_weighted2 = array(0, dim=c(1580, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 2:1580) {
lambdaarray_weighted2[k,i,j] = lambda_weighted2[i,j]
}}}

# The following are the predicted survival fractions for each i,j,k.
theoretical = array(0, dim=c(1580, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 2:1580) {
theoretical[k,i,j] =exp(-lambdaarray_weighted[k,i,j]*blIntegrals3[k, i, j])
}}}

# We are comparing groups A and B, where A corresponds to the Sokal group, and B
corresponds to one of the 45 mechanisms.

# 2x2 table for Sokal
n_Sokal = y[1:length(y[2:1580])]*1635 # number of people at risk at time i
y1 = y[2:length(y[2:1580])]
y2 = y[1:(length(y[2:1580])-1)]
d_Sokal  = (y2-y1)*1635 # number of people who die at time i
d_Sokal = c(0,d_Sokal)

# 2x2 table for mechanisms
n_mech = array(0, dim=c(1580, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 2:1580) {
n_mech[k, i,j] = theoretical[k , i, j]*1635 # number of people at risk at time k
}}}

theoretical_1 = array(0, dim=c(1580, 5, 6))
theoretical_2 = array(0, dim=c(1580, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 2:1579) {
theoretical_1[k, i, j] = theoretical[(k+1), i, j]
theoretical_2[k, i, j] = theoretical[k, i, j]
}}}

d_mech = array(0, dim=c(1580, 5, 6))
d_mech[1,i,j]=0
for (i in 1:5) { for (j in 1:6) { for (k in 2:1579) {
d_mech[(k+1),i,j] = (theoretical_2[(k+1), i, j]-theoretical_1[(k+1), i, j])*1635
}}}

# Expected number of deaths at each time for Sokal
# Creates the E_At=d(n_A/n), bottom of page 5 of surv.pdf
Et_Sokal = array(0, dim=c(1580, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 2:1580) {
Et_Sokal[k,i,j] = (n_Sokal[k])*(d_Sokal[k] + d_mech[k, i, j])/(n_Sokal[k] + n_mech[k, i,
j])
}}}
```

```
# Expected number of deaths in Sokal group
# E_A sum of E_At over all t, or #3 on page 6 of surv.pdf
E_Sokal = array(0, dim=c(5, 6))
for (i in 1:5) { for (j in 1:6) {
E_Sokal[i,j] = sum(Et_Sokal[2:1579,i,j])
}}

# Expected number of deaths at each time for mechanisms
# Creates the E_Bt=d(n_B/n), top of page 6 of surv.pdf
Et_mech = array(0, dim=c(1580, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 2:1580) {
Et_mech[k,i,j] = (n_mech[k, i, j])*(d_Sokal[k] + d_mech[k, i, j])/(n_Sokal[k] + n_mech[k,
i, j])
}}}

# Expected number of deaths in mechanisms group
# E_B sum of E_Bt over all t
E_mech = array(0, dim=c(5, 6))
for (i in 1:5) { for (j in 1:6) {
E_mech[i,j] = sum(Et_mech[2:1579,i,j])
}}

# Observed number of deaths in Sokal
# O_A=sum of the d_A over all the t, or #1 on page 6 of surv.pdf
O_Sokal = sum(d_Sokal[2:1579])
O_Sokal = array(O_Sokal, dim=c(5, 6))

# Observed number of deaths in mechanisms
# O_B=sum of the d_B over all the t
O_mech = array(0, dim=c(5, 6))
for (i in 1:5) { for (j in 1:6) {
O_mech[i,j] = sum(d_mech[2:1580,i,j])
}}

# Chi-square test statistic
# Equation (*) on page 6 of surv.pdf
chisq = (O_Sokal-E_Sokal)^2/E_Sokal + (O_mech-E_mech)^2/E_mech
df=1

# Repeat for the 5x5 array of leuk-leuk mechanisms.
theoretical2 = array(0, dim=c(1580, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 2:1580) {
theoretical2[k,i,j] =
exp(-lambdaarray_weighted2[k,i,j]*blIntegrals4[k, i, j])
}}}

# 2x2 table for Sokal
n_Sokal = y[1:length(y)]*1635 # number of people at risk at time i
y1 = y[2:length(y)]
y2 = y[1:(length(y)-1)]
d_Sokal  = (y2-y1)*1635 # number of people who die at time i
d_Sokal = c(0,d_Sokal)
n_mech2 = array(0, dim=c(1580, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 1:1580) {
n_mech2[k, i,j] = theoretical2[k , i, j]*1635 # number of people at risk at time k
}}}
theoretical2_1=array(0, dim=c(1580, 5, 5))
theoretical2_2=array(0, dim=c(1580, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 2:1579) {
theoretical2_1[k, i, j] = theoretical2[(k+1), i, j]
theoretical2_2[k, i, j] = theoretical2[k, i, j]
}}}
```

```
d_mech2 = array(0, dim=c(1580, 5, 5))
d_mech2[1,i,j]=0
for (i in 1:5) { for (j in i:5) { for (k in 1:1579) {
d_mech2[(k+1),i,j] = (theoretical2_2[(k+1), i, j]-theoretical2_1[(k+1), i, j])*1635
}}}
Et_Sokal2 = array(0, dim=c(1580, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 2:1580) {
Et_Sokal2[k,i,j] = (n_Sokal[k])*(d_Sokal[k] + d_mech2[k, i, j])/(n_Sokal[k] + n_mech2[k,
i, j])
}}}
E_Sokal2 = array(0, dim=c(5, 5))
for (i in 1:5) { for (j in i:5) {
E_Sokal2[i,j] = sum(Et_Sokal2[2:1580,i,j])
}}
Et_mech2 = array(0, dim=c(1580, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 2:1580) {
Et_mech2[k,i,j] = (n_mech2[k,i,j])*(d_Sokal[k] + d_mech2[k, i, j])/(n_Sokal[k] +
n_mech2[k, i, j])
}}}
E_mech2 = array(0, dim=c(5, 5))
for (i in 1:5) { for (j in i:5) {
E_mech2[i,j] = sum(Et_mech2[2:1580,i,j])
}}
O_Sokal2 = array(O_Sokal, dim=c(5,5))
O_mech2 = array(0, dim=c(5, 5))
for (i in 1:5) { for (j in i:5) {
O_mech2[i,j] = sum(d_mech2[2:1580,i,j])
}}
chisq2 = (O_Sokal2 - E_Sokal2)^2/E_Sokal2 + (O_mech2 - E_mech2)^2/E_mech2

# Save the p-values.
pval=pchisq(chisq, df=1, lower.tail=FALSE)
pval2=pchisq(chisq2, df=1, lower.tail=FALSE)

# Estimating lambda by minimizing sum of squares errors
yarray = array(0, dim=c(1581, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 1:1581) {
yarray[k,i,j] = y[k]
}}}
lambdaarray_weighted = array(0, dim=c(1581, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 1:1581) {
lambdaarray_weighted[k,i,j] = lambda_weighted[i,j]
}}}

# We know that the non-linear estimates are slightly greater than those from VWLS. We
want to find the multiplier such that the following holds:
# Non-linear estimate = VWLS estimate * multiplier.
# The multiplier should correspond to all 45 mechanisms, and always be at least 1.
# The values of the multiplier are saved in arrays multiple of dim=c(1581,5,6) and
multiple2 of dim=c(1581,5,5).
# Can see the multipliers using multiple[1,,] and multiple2[1,,]

multiple = array(0, dim=c(1581, 5,6))  # Array to multiply to lambda_weighted to obtain
min SSE estimates since we want to increase the lambda_weighted estimates slightly.
# The x a vector of possible multipliers. Here, we limit ourselves to integers 1 to 1001.
Based on running these lines many times, multiple and multiple2 need not exceed ~50 or
so, so 1001 is a good enough upper bound.

x=1+(0:1000)/1
# For each of the x, calculate the predicted survival fraction, calculate the sum of
squares error for each prediction. The goal is to find the entry of x = 1:1001 that gives
the smallest sum of squares error.
```

```
# Save the vector of sum of squares error into SSEvec.

SSEvec= array(0, dim=c(length(x), 5, 6))  # Vector of SSE for each entry in x
for (i in 1:5) { for (j in 1:6) { for (k in 1:length(x)) {
SSEvec[k,i,j]=sum((yarray[1:1581,i,j]-exp(-
x[k]*lambdaarray_weighted[1:1581,i,j]*blIntegrals3[1:1581,i,j]))^2)
}}}
for (i in 1:5) { for (j in 1:6) { for (k in 1:1581) {
multiple[k, i,j] =x[which(SSEvec[,i,j]==min(SSEvec[,i,j]))] # Choose the entry of x that
minimizes SSE
}}}
# multiple[1,,] # Used to see crude estimates of multiple[1,,]

# Now that we have crude estimates of the multipliers, we can refine these values by
testing their surrounding values, but instead of integers, we allow for precision to 3
decimal places.
# The new x, which is still the vector of possible multipliers, takes the previously
calculated crude value, and tests for close, surrounding values:

for (m in 1:5) { for (n in 1:6) {
x=multiple[1,m,n]+(-1000:1000)/1000 # For precision to 3 decimal places
# Again, these x entries are tested by having their sums of squares errors calculated:

SSEvec= array(0, dim=c(length(x), 5, 6))
for (i in m) { for (j in n) { for (k in 1:length(x)) {
SSEvec[k,i,j]=sum((yarray[1:1581,i,j]-exp(-
x[k]*lambdaarray_weighted[1:1581,i,j]*blIntegrals3[1:1581,i,j]))^2)
}}}
# Then we take the multiplier corresponding to the x entry that gave the smallest sum of
squares error:

for (k in 1:1581) { multiple[k, m,n]=min(x[which(SSEvec[,m,n]== min(SSEvec[,m,n]))] )
}}}
# multiple[1,,] # Used to see estimates of multiple[1,,] to 3 decimal places

# The values can be estimated to more places after the decimal.
# Repeat for the 5x5 matrix:
## leuk-leuk
lambdaarray_weighted2= array(0, dim=c(1581, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 1:1581) {
lambdaarray_weighted2[k,i,j] = lambda_weighted2[i,j]
}}}
multiple2 = array(1, dim=c(1581, 5, 5))

x=1+(0:1000)/1
SSEvec2= array(0, dim=c(length(x), 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 1:length(x)) {
SSEvec2[k,i,j]=sum((yarray[1:1581,i,j]-exp(-
x[k]*lambdaarray_weighted2[1:1581,i,j]*blIntegrals4[1:1581,i,j]))^2)
}}}

for (k in 1:1581) {
multiple2[k,1,1]=min(x[which(SSEvec2[,1,1]==min(SSEvec2[,1,1]))])
multiple2[k,1,2]=min(x[which(SSEvec2[,1,2]==min(SSEvec2[,1,2]))])
multiple2[k,1,3]=min(x[which(SSEvec2[,1,3]==min(SSEvec2[,1,3]))])
multiple2[k,1,4]=min(x[which(SSEvec2[,1,4]==min(SSEvec2[,1,4]))])
multiple2[k,1,5]=min(x[which(SSEvec2[,1,5]==min(SSEvec2[,1,5]))])
multiple2[k,2,2]=min(x[which(SSEvec2[,2,2]==min(SSEvec2[,2,2]))])
multiple2[k,2,3]=min(x[which(SSEvec2[,2,3]==min(SSEvec2[,2,3]))])
multiple2[k,2,4]=min(x[which(SSEvec2[,2,4]==min(SSEvec2[,2,4]))])
multiple2[k,2,5]=min(x[which(SSEvec2[,2,5]==min(SSEvec2[,2,5]))])
multiple2[k,3,3]=min(x[which(SSEvec2[,3,3]==min(SSEvec2[,3,3]))])
```

```
multiple2[k,3,4]=min(x[which(SSEvec2[,3,4]==min(SSEvec2[,3,4]))])
multiple2[k,3,5]=min(x[which(SSEvec2[,3,5]==min(SSEvec2[,3,5]))])
multiple2[k,4,4]=min(x[which(SSEvec2[,4,4]==min(SSEvec2[,4,4]))])
multiple2[k,4,5]=min(x[which(SSEvec2[,4,5]==min(SSEvec2[,4,5]))])
multiple2[k,5,5]=min(x[which(SSEvec2[,5,5]==min(SSEvec2[,5,5]))])}
# multiple2[1,,]

for (m in 1:5) { for (n in m:5) {
x=multiple2[1,m,n]+(-1000:1000)/1000
SSEvec2= array(0, dim=c(length(x), 5, 5))
for (i in m) { for (j in n) { for (k in 1:length(x)) {
SSEvec2[k,i,j]=sum((yarray[1:1581,i,j]-exp(-
x[k]*lambdaarray_weighted2[1:1581,i,j]*blIntegrals4[1:1581,i,j]))^2)
}}}
for (k in 1:1581) {
multiple2[k, m,n]=min(x[which(SSEvec2[,m,n]==min(SSEvec2[,m,n]))])
}}}
# multiple2[1,,]

# P-values from log-rank test, same code as before
yarray = array(0, dim=c(1581, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 1:1581) {
yarray[k,i,j] = y[k]
}}}
lambdaarray = array(0, dim=c(1581, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 1:1581) {
lambdaarray[k,i,j] = lambda_weighted[i,j]
}}}
lambdaarray2 = array(0, dim=c(1581, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 1:1581) {
lambdaarray2[k,i,j] = lambda_weighted2[i,j]
}}}
# One-cell, Normal-Leuk
theoretical_SSE = array(0, dim=c(1581, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 1:1581) {
theoretical_SSE[k,i,j] =
exp(-multiple[k,i,j]*lambdaarray_weighted[k,i,j]*blIntegrals3[k, i, j])
}}}

# 2x2 table for Sokal
n_Sokal = y[1:length(y[1:1581])]*1635
y1 = y[2:length(y[1:1581])]
y2 = y[1:(length(y[1:1581])-1)]
d_Sokal  = (y2-y1)*1635
d_Sokal = c(0,d_Sokal)
# 2x2 table for mechanisms
n_mech = array(0, dim=c(1581, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 1:1581) {
n_mech[k, i,j] = theoretical_SSE[k , i, j]*1635
}}}

theoretical_1 = array(0, dim=c(1581, 5, 6))
theoretical_2 = array(0, dim=c(1581, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 2:1579) {
theoretical_1[k, i, j] = theoretical_SSE[(k+1), i, j]
theoretical_2[k, i, j] = theoretical_SSE[k, i, j]
}}}

d_mech = array(0, dim=c(1581, 5, 6))
d_mech[1,i,j]=0
for (i in 1:5) { for (j in 1:6) { for (k in 2:1579) {
d_mech[(k+1),i,j] = (theoretical_2[(k+1), i, j]-theoretical_1[(k+1), i, j])*1635
```

```
}}}

Et_Sokal = array(0, dim=c(1581, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 1:1581) {
Et_Sokal[k,i,j] = (n_Sokal[k])*(d_Sokal[k] +
d_mech[k, i, j])/(n_Sokal[k] + n_mech[k, i, j])  }}}
E_Sokal = array(0, dim=c(5, 6))
for (i in 1:5) { for (j in 1:6) {
E_Sokal[i,j] = sum(Et_Sokal[2:1579,i,j])
}}

Et_mech = array(0, dim=c(1581, 5, 6))
for (i in 1:5) { for (j in 1:6) { for (k in 1:1581) {
Et_mech[k,i,j] = (n_mech[k, i, j])*(d_Sokal[k] +
d_mech[k, i, j])/(n_Sokal[k] + n_mech[k, i, j])   }}}
E_mech = array(0, dim=c(5, 6))
for (i in 1:5) { for (j in 1:6) {
E_mech[i,j] = sum(Et_mech[2:1579,i,j])
}}

O_Sokal = sum(d_Sokal[2:1579])
O_Sokal = array(O_Sokal, dim=c(5, 6))

O_mech = array(0, dim=c(5, 6))
for (i in 1:5) { for (j in 1:6) {
O_mech[i,j] = sum(d_mech[1:1581,i,j])
}}

chisq = (O_Sokal - E_Sokal)^2/E_Sokal + (O_mech -
E_mech)^2/E_mech
df=1
# Leuk-leuk
theoretical2_SSE = array(0, dim=c(1581, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 1:1581) {
theoretical2_SSE[k,i,j] =
exp(-multiple2[k,i,j]*lambdaarray2[k,i,j]*blIntegrals4[k, i, j])   }}}

n_Sokal = y[1:length(y)]*1635
y1 = y[2:length(y)]
y2 = y[1:(length(y)-1)]
d_Sokal  = (y2-y1)*1635
d_Sokal = c(0,d_Sokal)
n_mech2 = array(0, dim=c(1581, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 1:1581) {
n_mech2[k, i,j] = theoretical2_SSE[k , i, j]*1635
}}}
theoretical2_1=array(0, dim=c(1581, 5, 5))
theoretical2_2=array(0, dim=c(1581, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 2:1579) {
theoretical2_1[k, i, j] = theoretical2_SSE[(k+1), i, j]
theoretical2_2[k, i, j] = theoretical2_SSE[k, i, j]
}}}
d_mech2 = array(0, dim=c(1581, 5, 5))
d_mech2[1,i,j]=0
for (i in 1:5) { for (j in i:5) { for (k in 1:1579) {
d_mech2[(k+1),i,j] = (theoretical2_2[(k+1), i, j]-theoretical2_1[(k+1), i, j])*1635
}}}
Et_Sokal2 = array(0, dim=c(1581, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 1:1581) {
```

```
Et_Sokal2[k,i,j] = (n_Sokal[k])*(d_Sokal[k] + d_mech2[k, i, j])/(n_Sokal[k] + n_mech2[k,
i, j])
}}}
E_Sokal2 = array(0, dim=c(5, 5))
for (i in 1:5) { for (j in i:5) {
E_Sokal2[i,j] = sum(Et_Sokal2[1:1581,i,j])
}}
Et_mech2 = array(0, dim=c(1581, 5, 5))
for (i in 1:5) { for (j in i:5) { for (k in 1:1581) {
Et_mech2[k,i,j] = (n_mech2[k,i,j])*(d_Sokal[k] + d_mech2[k, i, j])/(n_Sokal[k] +
n_mech2[k, i, j])
}}}
E_mech2 = array(0, dim=c(5, 5))
for (i in 1:5) { for (j in i:5) {
E_mech2[i,j] = sum(Et_mech2[1:1581,i,j])
}}
O_Sokal2 = array(O_Sokal, dim=c(5,5))
O_mech2 = array(0, dim=c(5, 5))
for (i in 1:5) { for (j in i:5) {
O_mech2[i,j] = sum(d_mech2[1:1581,i,j])
}}
chisq2 = (O_Sokal2 - E_Sokal2)^2/E_Sokal2 +
(O_mech2 - E_mech2)^2/E_mech2
df=1

pval_SSE=pchisq(chisq, df=1, lower.tail=FALSE)
pval2_SSE=pchisq(chisq2, df=1, lower.tail=FALSE)
##********* Outputting more results********##
## record parameters
values = c('alpha' = alpha, 'beta' = beta, 'd0' = d0, 'd1' = d1, 'd2' = d2, 'd3' = d3,
'rx' = rx, 'ry' = ry, 'rytr' = rytr, 'px' = px, 'py' = py, 'ax' = ax, 'ay' = ay, 'aytr' =
aytr, 'by' = by, 'bytr' = bytr, 'cx' = cx, 'cy' = cy, 'ttr'=ttr,'cytr' = cytr, 'L'=L)
write(c(paste(names(values),'=',values),'\n'),file = 'Parameters.txt')

eql = c('x0eq' = x0eq, 'y0eq' = y0eq, 'x1eq' = x1eq, 'x2eq'= x2eq, 'x3eq' = x3eq)
write(c(paste(names(eql),'=',eql),'\n'),file = 'Parameters.txt', append = TRUE)

rownames=c("yq","y0","y1","y2","y3")
colnames1=c("1","xq","x0","x1","x2","x3")
colnames2=rownames

write("multiple[1,,]", file="output.txt")
write.table(multiple[1,,], file = "output.txt", sep = " ", append=TRUE,
row.names=rownames, col.names=colnames1
)
write("\n", file="output.txt", append=TRUE)

write("multiple2[1,,]", file="output.txt", append=TRUE)
write.table(multiple2[1,,], file = "output.txt", sep = " ", append=TRUE,
row.names=rownames, col.names=colnames2
)
write("\n", file="output.txt", append=TRUE)

write("lambda_weighted", file="output.txt", append=TRUE)
write.table(lambda_weighted, file = "output.txt", sep = " ", append=TRUE,
row.names=rownames, col.names=colnames1
)
```

```
write("\n", file="output.txt", append=TRUE)

write("lambda_weighted2", file="output.txt", append=TRUE, )
write.table(lambda_weighted2, file = "output.txt", sep = " ", append=TRUE,
row.names=rownames, col.names=colnames2
)
write("\n", file="output.txt", append=TRUE)

write("pval", file="output.txt", append=TRUE)
write.table(pval, file = "output.txt", sep = " ", append=TRUE,
row.names=rownames, col.names=colnames1
)
write("\n", file="output.txt", append=TRUE)

write("pval2", file="output.txt", append=TRUE)
write.table(pval2, file = "output.txt", sep = " ", append=TRUE,
row.names=rownames, col.names=colnames2
)
write("\n", file="output.txt", append=TRUE)

write("lambda_SSE", file="output.txt", append=TRUE)
write.table(lambda_weighted*multiple[1,,], file = "output.txt", sep = " ", append=TRUE,
row.names=rownames, col.names=colnames1
)
write("\n", file="output.txt", append=TRUE)

write("lambda_SSE2", file="output.txt", append=TRUE)
write.table(
lambda_weighted2*multiple2[1,,], file = "output.txt", sep = " ", append=TRUE,
row.names=rownames, col.names=colnames2
)
write("\n", file="output.txt", append=TRUE)

write("pval_SSE", file="output.txt", append=TRUE)
write.table(pval_SSE, file = "output.txt", sep = " ", append=TRUE,
row.names=rownames, col.names=colnames1
)
write("\n", file="output.txt", append=TRUE)

write("pval2_SSE", file="output.txt", append=TRUE)
write.table(pval2_SSE, file = "output.txt", sep = " ", append=TRUE,
row.names=rownames, col.names=colnames2
)
# ## *******Plotting Best Fits.*********##
##In the following adjust i and j using pval_SSE and pval2_SSE output above to get high
p-values
pdf_name= c('bestfits.pdf')

pdf(file = pdf_name, height=5, width=9)

# pdf(file = 'bestfits.pdf', height=5, width=9)


par(mfrow=c(1,2))

plot(1:1581/109, y[1:1581], type='l',
# main="Representative Fits \n Minimizing SSE",
xlab='Time since clinical presentation (years)',
ylab='Survival fraction', ylim=c(0,1))
# Leuk-normal
for (i in 2) { for (j in 3) {
points(1:1581/109,exp(-
```

```
multiple[1:1581,i,j]*lambdaarray_weighted[1:1581,i,j]*blIntegrals3[1:1581,
i,j]), type='l', col='red')
}}
# One-cell
for (i in 2) { for (j in 1) {
points(1:1581/109,
exp(-multiple[1:1581,i,j]*lambdaarray_weighted[1:1581,i,j]*blIntegrals3[1:1581, i,j]),
type='l', col='blue')
}}
# Leukemic-Leukemic
for (i in 2) { for (j in 2) {
points(1:1581/109, exp(-
multiple2[1:1581,i,j]*lambdaarray_weighted2[1:1581,i,j]*blIntegrals4[1:1581, i,j]),
type='l', col='green')
}}
points(1:1581/109, y[1:1581], type='l')
legend ("topright", c("One-Cell", "Leukemic-Normal", "Leukemic-Leukemic", "Sokal"),
fill=c("blue", "red", "green", "black"), cex=.70)
# Plot 2 - Semi-log
plot(1:1581/109, log(y[1:1581]), type='l',
#main="Representative Fits \nMinimizing SSE \nSemi-log",
xlab='Time since clinical presentation (years)', ylab='Log(Survival fraction)',ylim=c(-
3.5, 0))
# Leuk-normal
for (i in 2) { for (j in 3) {
points(1:1581/109,
(-multiple[1:1581,i,j]*lambdaarray_weighted [1:1581,i,j]*blIntegrals3[1:1581, i,j]),
type='l', col='red')
}}
# One-cell
for (i in 2) { for (j in 1) {
points(1:1581/109,
(-multiple[1:1581,i,j]*lambdaarray_weighted [1:1581,i,j]*blIntegrals3[1:1581,i,j]),
type='l', col='blue')
}}
# Leukemic-Leukemic
for (i in 2) { for (j in 2) {
points(1:1581/109,
(-multiple2[1:1581,i,j]*lambdaarray_weighted2[1:1581,i,j]*blIntegrals4[1:1581, i,j]),
type='l', col='green')
}}

points(1:1581/109, log(y[1:1581]), type='l')

legend ("topright", c("One-Cell", "Leukemic-Normal", "Leukemic-Leukemic", "Sokal"),
fill=c("blue", "red", "green", "black"), cex=.70)

dev.off()
```

Next: S7. Or jump to:

S1. Robustness relative to changes in the stem cell proliferation parameter
S2. Stochastic effects
S3. Statistical methods.
S4. Response of main calculation to changes in the blast crisis latency time *L_BC*
S5. Response of main calculation to changes in stem cell transition parameters
S6. Customized computer program used
S7. Supplement bibliography

S7. Supplement bibliography

The bibilography is compiled seperately from that for the main text. It has some overlap.

- Almeida AM, Castel-Branco MM & Falcao AC, Linear regression for calibration lines revisited: weighting schemes for bioanalytical methods. J Chromatogr B Analyt Technol Biomed Life Sci 774:215-22, 2002.
- Berkson J & Gage RP, Survival Curve for Cancer Patients Following Treatment. Journal of the American Statistical Association 471952.
- Buckley J & James I, Linear Regression with Censored Data. Biometrika 66:429-436, 1979.
- Cacoullos T, The F-test of homoscedasticity for correlated normal variables. STATISTICS AND PROBABILITY LETTERS 54:1-3, 2001.
- Devore J, 1995, *Probabilty and Statistics for Engineering and the Sciences (4thEd)*, ITP, NY,
- Fakir H, Tan W, Hlatky L, Hahnfeldt P & Sachs R, Stochastic population dynamic effects for lung cancer progression. Radiat Res Suppl 172:383-93, 2009.
- Foo J, Drummond MW, Clarkson B, Holyoake T & Michor F, Eradication of chronic myeloid leukemia stem cells: a novel mathematical model predicts no therapeutic benefit of adding G-CSF to imatinib. PLoS Comput Biol 5:e1000503, 2009.
- Kim PS, Lee PP & Levy D, Modeling imatinib-treated chronic myelogenous leukemia: reducing the complexity of agent-based models. Bull Math Biol 70:728-44, 2008.
- Lenaerts T, Pacheco JM, Traulsen A & Dingli D, Tyrosine kinase inhibitor therapy can cure chronic myeloid leukemia without hitting leukemic stem cells. Haematologica 2009.
- Liefers GJ, Cleton-Jansen AM, van de Velde CJ, Hermans J, van Krieken JH, Cornelisse CJ & Tollenaar RA, Micrometastases and survival in stage II colorectal cancer. N Engl J Med 339:223-8, 1998.
- Little MP, Weiss HA, Boice JD, Jr., Darby SC, Day NE & Muirhead CR, Risks of leukemia in Japanese atomic bomb survivors, in women treated for cervical cancer, and in patients treated for ankylosing spondylitis. Radiat Res 152:280-92, 1999.
- Little MP, Hoel DG, Molitor J, Boice JD, Wakeford R & Muirhead CR, New models for evaluation of radiation-induced lifetime cancer risk and its uncertainty employed in the UNSCEAR 2006 report. Radiat Res 169:660-76, 2008.
- Ma S, Principal Component Analysis in Linear Regression Survival Model with Microarray Data. Journal of Data Science 5:183-98, 2007.
- Michor F, Hughes TP, Iwasa Y, Branford S, Shah NP, Sawyers CL & Nowak MA, Dynamics of chronic myeloid leukaemia. Nature 435:1267-70, 2005.
- Orlic D, Kajstura J, Chimenti S, Limana F, Jakoniuk I, Quaini F, Nadal-Ginard B, Bodine DM, Leri A, et al., Mobilized bone marrow cells repair the infarcted heart, improving function and survival. Proc Natl Acad Sci U S A 98:10344-9, 2001.
- Preston DL, Kusumi S, Tomonaga M, Izumi S, Ron E, Kuramoto A, Kamada N, Dohy H, Matsuo T, et al., Cancer incidence in atomic bomb survivors. Part III. Leukemia, lymphoma and multiple myeloma, 1950-1987. Radiat Res 137:S68-97, 1994.
- Radivoyevitch T & Hoel DG, Biologically-based risk estimation for radiation-induced chronic myeloid leukemia. Radiation and Environmental Biophysics 39:153-159, 2000.
- Radivoyevitch T, Kozubek S & Sachs R, Biologically based risk estimation for radiation-induced CML - Inferences from BCR and ABL geometric distributions. Radiation and Environmental Biophysics 40:1-9, 2001.
- Richardson D, Sugiyama H, Nishi N, Sakata R, Shimizu Y, Grant EJ, Soda M, Hsu WL, Suyama A, et al., Ionizing radiation and leukemia mortality among Japanese Atomic Bomb Survivors, 1950-2000. Radiat Res 172:368-82, 2009.
- Roeder I, Horn M, Glauche I, Hochhaus A, Mueller MC & Loeffler M, Dynamic modeling of imatinib-treated chronic myeloid leukemia: functional insights and clinical implications. Nat Med 12:1181-4, 2006.
- Sokal JE, Prognosis in chronic myeloid leukaemia: biology of the disease vs. treatment. Baillieres Clin Haematol 1:907-29, 1987.

Jump to: