Quantitative RT - PCR Analysis Program

- **The program is written in Mathematica 7.0.0**
- **See Hernandez M.C. and Janusonis S. (2010) for details. The program is based, in part, on Tichopad A. et al. (2003) Nucleic Acids Research 31 : e122 (note: the typos in Tichopad et al. have been corrected).**

```
(* Five errors and five warnings can be reported. *)

MaxErrorN = 5;
MaxWarningN = 5;

(* The number of results to be reported for each amplification. *)

ReportDataN = 16;

(* IMPORT DATA *)
(* All data should be obtained with the same
 PCR primers and under the same cycling conditions,
but different wells may represent different experimental conditions
 and different plates. The first column contains the cycle numbers;
the other columns contain the raw fluorescence values of each
 amplification (e.g., 60 wells with 45 cycles should be
    represented by a 45 X 61 matrix with no column or row names). *)

AllResults = Flatten[Import["C:\FilePath\FileName.xls"], 1];
DR = Dimensions[AllResults];
ReportDimensionError = 0;
If[DR[[1]] < 20, ReportDimensionError = 1];
(* Report "Error 1" if too few cycles to begin analysis. *)
AllResultsTr = Transpose[AllResults];

ReportResults = Table[0., {DR[[2]]}, {ReportDataN}];

ReportResults[[1, 1]] = "Well";
ReportResults[[1, 2]] = "mRNA*10^6";
```

```
ReportResults[[1, 3]] = "E";
ReportResults[[1, 4]] = "FixedE";
ReportResults[[1, 5]] = "X-intercept";
ReportResults[[1, 6]] = "BaselineMean";
ReportResults[[1, 7]] = "BaselineSlope";
ReportResults[[1, 8]] = "StartPoint";
ReportResults[[1, 9]] = "MeasurePoint";
ReportResults[[1, 10]] = "EndPoint";
ReportResults[[1, 11]] = "RegressionPoints";
ReportResults[[1, 12]] = "PointsRemaining";
ReportResults[[1, 13]] = "RegressionCorr";
ReportResults[[1, 14]] = "UserCriterion";
ReportResults[[1, 15]] = "Errors";
ReportResults[[1, 16]] = "Warnings";

ReportPlots = Table[0., {DR[[2]]}, {2}];

ReportPlots[[1, 1]] = "RFU(cycle)";
ReportPlots[[1, 2]] = "Log[RFU(cycle)-Baseline]";

For[WellC = 1, WellC ≤ DR[[2]] - 1,
 Clear[ReportError];
 Clear[ReportWarning];
 ReportError = Table[0, {5}];
 ReportWarning = Table[0, {5}];
 ReportError[[1]] = ReportDimensionError;

 Print["Well = ", WellC];

 Clear[Results];
 Results = MapThread[List, {AllResultsTr[[1]], AllResultsTr[[WellC + 1]]}];

 (* Variables and settings. *)

 InitialRNA = 0.;
 Efficiency = 0.;

 (* EfficiencyFixed is used to calculate mRNA amounts. Run the program,
 determine the mean efficiency, and enter it as EfficiencyFixed. *)

 EfficiencyFixed = 1.62297;
 XIntercept = 0.;
```

```
BackgroundMean = 0.;
BackSlope = 0.;
StartPoint = 0;
MeasurePoint = 0;
EndPoint = 0;
CorrFullSet = 0.;
Acceptable = 0;
TotalCycles = Length[Results];

SigPStart = 0.01; (* Significant studentized residual for StartPoint *)
SigPEnd = 0.10; (* Significant studentized residual for End Point *)
SigPStartN = 3; (* The number of consecutive significant points for StartPoint *)
SigPEndN = 1; (* The number of consecutive significant points for EndPoint *)
BackgroundRewind = 1;
(* The difference in steps between the last background point and StartPoint. *)
BackgroundCycles = 10; (* The number of points used for background calculations. *)

(* Get externally studentized residuals and their two-tailed P values *)

If[Plus @@ ReportError == 0,
 Clear[StudResPStart];
 StudResPStart = Table[1., {TotalCycles}];
 FirstCycle = 1;
 For[CC = FirstCycle + 3, CC ≤ TotalCycles,
  CycleN = CC - FirstCycle + 1;
  Clear[FirstPoints];
  Clear[FirstPointsMinus];
  FirstPoints = Take[Results, {FirstCycle, CC}];
  FirstPointsMinus = Take[Results, {FirstCycle, CC - 1}];
  MeanCycle = Mean[Transpose[FirstPoints][[1]]];
  Clear[linregminus];
  Clear[linreg];
  linreg[x_] = Normal[LinearModelFit[FirstPoints, x, x]];
  linregminus[x_] = Normal[LinearModelFit[FirstPointsMinus, x, x]];
```

$$h = \frac{1.}{\text{CycleN}} + \frac{(\text{FirstPoints}[[\text{CycleN}, 1]] - \text{MeanCycle})^2}{\text{Sum}\left[(\text{FirstPoints}[[i, 1]] - \text{MeanCycle})^2, \{i, 1, \text{CycleN}\}\right]};$$

$$\text{MS} = \text{Sum}\left[(\text{FirstPointsMinus}[[i, 2]] - \text{linregminus}[\text{FirstPointsMinus}[[i, 1]]])^2, \{i, 1, \text{CycleN} - 1\}\right] \Big/ (\text{CycleN} - 3);$$

```
  r = (FirstPoints[[CycleN, 2]] - linreg[FirstPoints[[CycleN, 1]]]) / Sqrt[MS * (1 - h)];
  StudResPStart[[CC]] = 2 * (1 - CDF[StudentTDistribution[CycleN - 2], Abs[r]]);
```

```
  CC++]];

(* Find the first exponential cycle based on externally studentized
  residuals. Note that more than one significant value is needed if SigPStartN >
 1. Report "Error 10" if fewer than 4 cycles are left when starting with StartPoint. *)

If[Plus @@ ReportError == 0,
 StartPointTrack = 3;
 Verification = 0;
 While[(Verification < SigPStartN) && (TotalCycles - StartPointTrack + SigPStartN ≥ 4),
   ++StartPointTrack;
   If[StudResPStart[[StartPointTrack]] ≥ SigPStart, Verification = 0];
   If[StudResPStart[[StartPointTrack]] < SigPStart, ++Verification];
  ]
  If[TotalCycles - StartPointTrack + SigPStartN < 4, ReportError[[2]] = 1];
 StartPoint = StartPointTrack - SigPStartN + 1
];

(* Find the mean background value. Report
 "Warning 1" if fewer points used than expected. *)
(* Report background slope = 0 if it is not significantly
  different (at α = 0.05) from zero. *)

If[Plus @@ ReportError == 0,
 FirstBackgroundPoint = Max[1, StartPoint - BackgroundRewind - BackgroundCycles + 1];
 LastBackgroundPoint = Max[1, StartPoint - BackgroundRewind];
 If[LastBackgroundPoint - FirstBackgroundPoint + 1 < BackgroundCycles,
  ReportWarning[[1]] = 1];
 BackgroundSet = Take[Results, {FirstBackgroundPoint, LastBackgroundPoint}];
 BackgroundMean = Mean[Transpose[BackgroundSet][[2]]];
 backreg = LinearModelFit[BackgroundSet, x, x, ConfidenceLevel → .95];
 backK1 = backreg["ParameterConfidenceIntervals"][[2, 1]];
 backK2 = backreg["ParameterConfidenceIntervals"][[2, 2]];
 If[backK1 ≤ 0 ≤ backK2, BackSlope = 0, BackSlope = backreg[[1, 2, 2]]]
];

(* If necessary, shift StartPoint forward ("Warning 100") so that only positive baseline-
 adjusted values are left. Report "Error 100" if after the shift
  fewer than 4 cycles are left when starting with StartPoint. *)

If[Plus @@ ReportError == 0,
 NegPass = 0;
```

```
 NegInd = 0;
 While[(NegPass == 0) && (StartPoint ≤ TotalCycles),
  For[i = StartPoint, i ≤ TotalCycles,
   If[Results[[i, 2]] - BackgroundMean <= 0, NegInd = 1];
   i++];
  If[NegInd == 1, ++StartPoint; NegInd = 0; ReportWarning[[3]] = 1, NegPass = 1];
 ]
];
If[TotalCycles - StartPoint + 1 < 4, ReportError[[3]] = 1];


(* Background-adjust and log-transform the data. *)

If[Plus @@ ReportError == 0,
 ResultsLog = MapThread[List,
   {Transpose[Results][[1]], Log[Transpose[Results][[2]] - BackgroundMean]}];
 Clear[StudResPEnd];
 StudResPEnd = Table[1., {TotalCycles}];
 FirstCycle = StartPoint;
 For[CC = FirstCycle + 3, CC ≤ TotalCycles,
  CycleN = CC - FirstCycle + 1;
  Clear[FirstPoints];
  Clear[FirstPointsMinus];
  FirstPoints = Take[ResultsLog, {FirstCycle, CC}];
  FirstPointsMinus = Take[ResultsLog, {FirstCycle, CC - 1}];
  MeanCycle = Mean[Transpose[FirstPoints][[1]]];
  Clear[loglinregminus];
  Clear[loglinreg];
  loglinreg[x_] = Normal[LinearModelFit[FirstPoints, x, x]];
  loglinregminus[x_] = Normal[LinearModelFit[FirstPointsMinus, x, x]];
```

$$h = \frac{1.}{\text{CycleN}} + \frac{(\text{FirstPoints}[[\text{CycleN}, 1]] - \text{MeanCycle})^2}{\text{Sum}\left[(\text{FirstPoints}[[i, 1]] - \text{MeanCycle})^2, \{i, 1, \text{CycleN}\}\right]};$$

$$\text{MS} = \text{Sum}\left[(\text{FirstPointsMinus}[[i, 2]] - \text{loglinregminus}[\text{FirstPointsMinus}[[i, 1]]])^2, \{i, 1, \text{CycleN} - 1\}\right] \Big/ (\text{CycleN} - 3);$$

```
  r = (FirstPoints[[CycleN, 2]] - loglinreg[FirstPoints[[CycleN, 1]]]) / Sqrt[MS * (1 - h)];
  StudResPEnd[[CC]] = 2 * (1 - CDF[StudentTDistribution[CycleN - 2], Abs[r]]);
  CC++]
];
```

```
(* Find the last exponential cycle based on externally studentized
   residuals. Note that more than one significant value is needed if SigPEndN >
  1. Report "Warning 1000" if the end of the data is reached. Notice EndPoint =
 EndPointTrack - Verification which covers both the normal and early termination. *)

If[Plus @@ ReportError == 0,
 EndPointTrack = StartPoint + 3;
 Verification = 0;
 While[(Verification < SigPEndN) && (EndPointTrack ≤ TotalCycles),
   If[StudResPEnd[[EndPointTrack]] ≥ SigPEnd, Verification = 0];
   If[StudResPEnd[[EndPointTrack]] < SigPEnd, ++Verification];
   ++EndPointTrack
  ]
  If[EndPointTrack > TotalCycles, ReportWarning[[4]] = 1; EndPointTrack = TotalCycles];
 EndPoint = EndPointTrack - Verification;
];

(* Do final calculations *)


If[Plus @@ ReportError == 0,

 UsePoints = Take[ResultsLog, {StartPoint, EndPoint}];
 RegSol = loglinreg[x_] = Normal[LinearModelFit[UsePoints, x, x]];
 Efficiency = Exp[RegSol[[2, 1]]];

 XIntercept = -RegSol[[1]] / RegSol[[2, 1]];

 FullSet = Take[ResultsLog, {StartPoint, EndPoint}];
 CorrFullSet = Correlation[Transpose[FullSet][[1]], Transpose[FullSet][[2]]];

 MeasurePoint = Round[(StartPoint + EndPoint) / 2.];
 InitialRNA = (Results[[MeasurePoint, 2]] - BackgroundMean) /
   (EfficiencyFixed ^Results[[MeasurePoint, 1]])
];

ReportResults[[WellC + 1, 1]] = WellC;
ReportResults[[WellC + 1, 2]] = InitialRNA * 10^6;
ReportResults[[WellC + 1, 3]] = Efficiency;
ReportResults[[WellC + 1, 4]] = EfficiencyFixed;
ReportResults[[WellC + 1, 5]] = XIntercept;
ReportResults[[WellC + 1, 6]] = BackgroundMean;
ReportResults[[WellC + 1, 7]] = BackSlope;
```

```
ReportResults[[WellC + 1, 8]] = StartPoint;
ReportResults[[WellC + 1, 9]] = MeasurePoint;
ReportResults[[WellC + 1, 10]] = EndPoint;
ReportResults[[WellC + 1, 11]] = EndPoint - StartPoint + 1;
ReportResults[[WellC + 1, 12]] = TotalCycles - EndPoint;
ReportResults[[WellC + 1, 13]] = CorrFullSet;

(* This criterion can be set by the user. It was not actually used in the Hernandez &
 Janusonis (2010) analysis. *)

If[(CorrFullSet > 0.99) && (EndPoint - StartPoint + 1 ≥ 5) && (TotalCycles - EndPoint ≥ 5),
 Acceptable = 1, Acceptable = 0];
ReportResults[[WellC + 1, 14]] = Acceptable;

ReportResults[[WellC + 1, 15]] = Sum[(10^(m - 1)) * ReportError[[m]], {m, 1, MaxErrorN}];
ReportResults[[WellC + 1, 16]] = Sum[(10^(m - 1)) * ReportWarning[[m]], {m, 1, MaxWarningN}];

(* Save the raw and Log plots. *)

ReportPlots[[WellC + 1, 1]] =
 ListPlot[Results, AxesOrigin → {0, 0}, PlotRange → {0, 1000}, AxesLabel → WellC];

If[Plus @@ ReportError == 0,

 ReportPlots[[WellC + 1, 2]] =
  Show[ListPlot[ResultsLog, AxesOrigin → {0, 0}, PlotRange → {-4, 6}],
   ListPlot[Table[{ResultsLog[[i, 1]], loglinreg[ResultsLog[[i, 1]]]}, {i, 1, 40}],
    Joined → True, AxesOrigin → {0, 0}, PlotRange → {-4, 6}],
   ListPlot[UsePoints, PlotStyle → PointSize[0.03],
    AxesOrigin → {0, 0}, PlotRange → {-4, 6}]],

 (* Otherwise make an empty plot. *)

 ListPlot[{{0, 0}}, AxesOrigin → {0, 0}, PlotRange → {-4, 6}]
];

WellC++;
]
```

```
Well = 1

Well = 2

Well = 3

Well = 4

Well = 5

Well = 6

Well = 7

Well = 8

Well = 9
```

```
Well = 10
Well = 11
Well = 12
Well = 13
Well = 14
Well = 15
Well = 16
Well = 17
Well = 18
Well = 19
Well = 20
Well = 21
Well = 22
Well = 23
Well = 24
Well = 25
Well = 26
Well = 27
Well = 28
Well = 29
Well = 30
Well = 31
Well = 32
Well = 33
Well = 34
Well = 35
Well = 36
Well = 37
Well = 38
Well = 39
Well = 40
Well = 41
Well = 42
Well = 43
Well = 44
```

```
Well = 45
Well = 46
Well = 47
Well = 48
Well = 49
Well = 50
Well = 51
Well = 52
Well = 53
Well = 54
Well = 55
Well = 56
Well = 57
Well = 58
Well = 59
Well = 60
Well = 61
Well = 62
Well = 63
Well = 64
Well = 65
Well = 66
```

ReportResults // TableForm

Well	mRNA*10^6	E	FixedE	X-intercept	BaselineMean	BaselineSlope	StartPoint	MeasurePc
1	1.15104	1.63279	1.62297	28.4305	156.113	0.470044	33	36
2	1.88229	1.76873	1.62297	28.2798	172.595	0.589576	32	34
3	1.05866	1.69299	1.62297	29.1092	164.07	0.533201	34	36
4	7.29052	1.70284	1.62297	25.1753	157.695	0.678505	30	32
5	29.0854	1.66398	1.62297	22.0864	172.547	1.0601	28	30
6	32.3477	1.79183	1.62297	22.8489	173.093	0.785524	28	30
7	33.1447	1.74778	1.62297	22.2912	203.568	0.987169	27	29

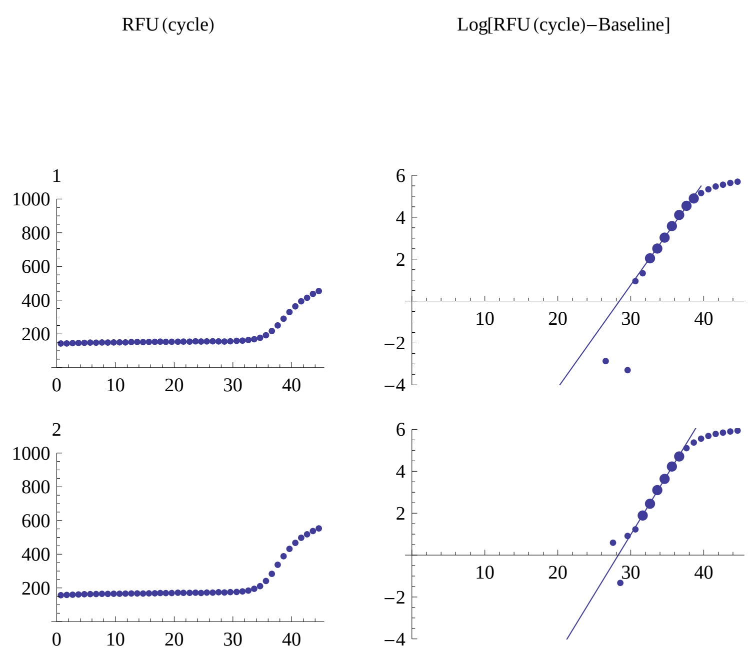
8	3.47205	1.5615	1.62297	25.5045	149.725	0.691226	31	32
9	1.34224	1.1786	1.62297	11.1109	182.733	0.755668	30	38
10	4.55521	1.64092	1.62297	25.612	175.443	0.481856	30	32
11	2.93863	1.59046	1.62297	26.1248	171.912	0.588467	31	33
12	2.7427	1.43319	1.62297	24.1122	199.322	0.78735	31	34
13	3.36266	1.56971	1.62297	25.6796	203.526	0.641116	30	32
14	0.375019	1.51376	1.62297	30.0298	171.826	0.245669	33	34
15	0.96664	1.26885	1.62297	24.0908	206.668	0.39637	31	33
16	2.04231	1.69737	1.62297	27.7321	162.053	0.688869	32	34
17	3.60176	1.62583	1.62297	26.0455	187.675	0.611625	31	34
18	3.40562	1.68137	1.62297	26.5849	178.344	0.550084	31	34
19	5.47887	1.69007	1.62297	25.5028	171.047	0.658833	30	32
20	5.66377	1.67133	1.62297	25.2233	202.257	1.019	30	32
21	15.2643	1.67629	1.62297	23.5687	177.643	0.639664	28	31
22	2.85976	1.61642	1.62297	26.3192	173.388	0.365058	30	34
23	12.785	1.71067	1.62297	23.9502	202.999	0.637793	28	30
24	0.68731	1.68975	1.62297	29.7648	185.435	0.752755	34	36
25	0.760616	1.79044	1.62297	30.2438	181.612	0.499182	34	36
26	5.10403	1.72999	1.62297	26.261	168.737	0.534349	30	33
27	5.8322	1.61476	1.62297	24.8574	203.881	1.11748	30	32
28	7.44736	1.63823	1.62297	24.5468	209.011	0.887722	29	32
29	7.87666	1.64503	1.62297	24.5999	185.424	0.827547	29	32
30	11.5644	1.59157	1.62297	23.2802	181.536	0.526718	27	31
31	14.2362	1.62543	1.62297	23.2961	214.438	0.857282	28	32
32	1.22528	1.53646	1.62297	27.3703	140.631	0.491433	32	36
33	1.20765	1.65418	1.62297	28.65	191.43	0	33	36
34	0.803076	1.62611	1.62297	29.2335	157.199	0.559527	34	37
35	2.66057	1.72654	1.62297	27.4953	149.353	0.199558	31	34
36	1.09892	1.55781	1.62297	27.7603	193.236	1.04559	33	36
37	5.64582	1.62092	1.62297	25.007	194.91	0.555614	29	33
38	4.80495	1.70519	1.62297	25.9142	160.442	0.506679	30	32
39	5.02038	1.61153	1.62297	25.1116	163.595	0	28	32
40	9.63597	1.62209	1.62297	23.9183	200.067	0.973932	29	32
41	2.19337	1.59896	1.62297	26.8332	137.967	0.482543	31	34
42	1.9939	1.59904	1.62297	26.9964	173.449	0.727219	32	35
43	2.49607	1.67445	1.62297	27.1187	161.658	0.649839	32	34
44	3.08664	1.59279	1.62297	25.7739	138.205	0.609191	30	32
45	3.6634	1.61267	1.62297	25.8186	179.832	0.650261	30	33
46	4.99723	1.51259	1.62297	24.1501	186.185	0	28	33
47	7.2101	1.58374	1.62297	24.0869	160.286	0.468265	29	32
48	11.3724	1.57923	1.62297	23.129	152.166	0.349515	27	30
49	12.0454	1.50956	1.62297	22.1128	186.942	0.534532	27	32
50	0.355251	1.60176	1.62297	30.4578	195.798	0.49736	35	38
51	0.53462	1.60245	1.62297	29.7181	178.014	0.401305	34	38
52	0.847537	1.65868	1.62297	29.2746	168.935	0.291581	33	36

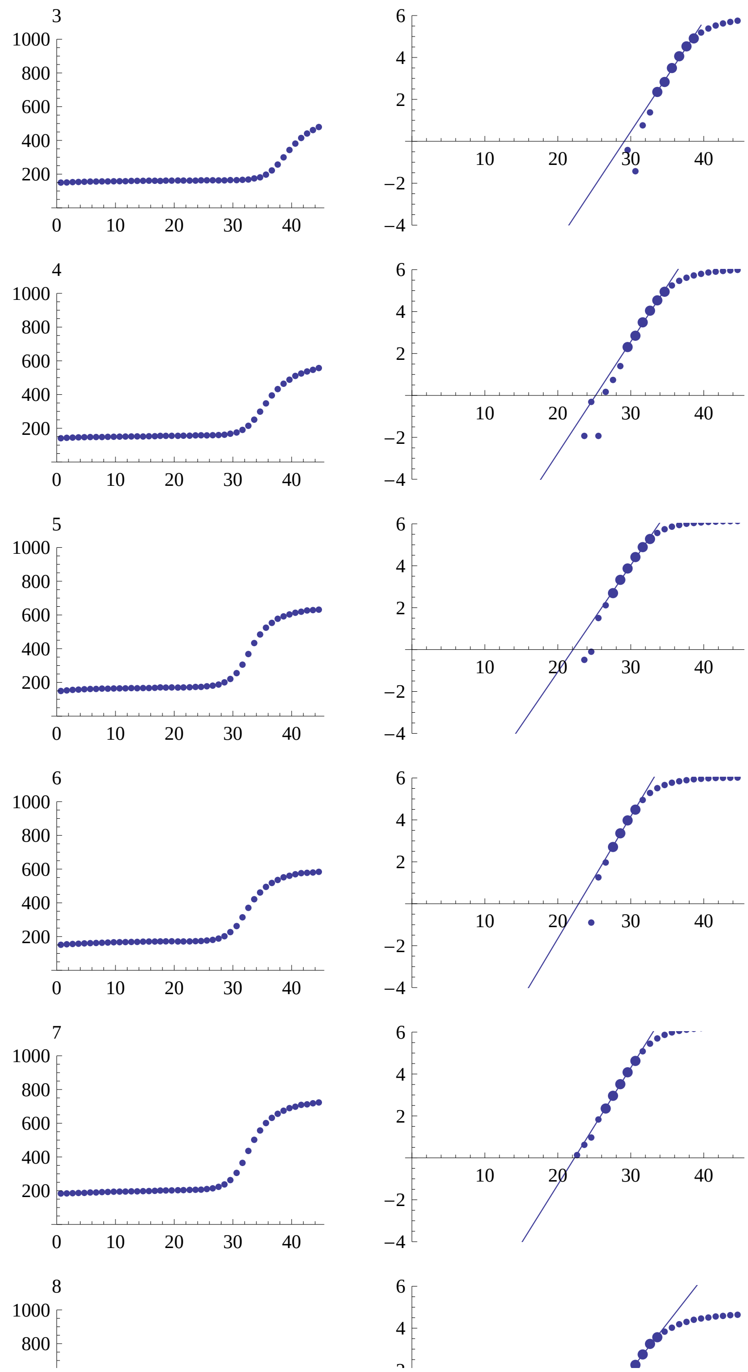
53	1.251	1.68791	1.62297	28.6457	212.07	0.937058	33	36
54	2.21814	1.58684	1.62297	26.7236	213.53	0.678959	31	36
55	5.83073	1.60356	1.62297	24.8654	181.181	0.872458	30	34
56	5.38374	1.58457	1.62297	24.6278	185.234	0.422514	28	32
57	8.30804	1.58826	1.62297	23.876	219.534	0.870563	29	30
58	0.726965	1.58518	1.62297	28.8587	147.02	0.675939	34	36
59	0.727639	1.65655	1.62297	29.4567	167.989	0.609827	34	36
60	1.47572	1.77806	1.62297	28.701	158.022	0.51059	33	34
61	2.3407	1.50315	1.62297	25.8743	147.515	0.283273	30	34
62	7.38187	1.7243	1.62297	25.3213	183.425	1.09425	30	32
63	7.75346	1.65019	1.62297	24.5968	185.867	0.756758	29	32
64	7.84122	1.58387	1.62297	23.9203	150.953	0.582736	28	30
65	8.74774	1.68221	1.62297	24.7238	150.384	0.532886	29	32
66	10.6464	1.66213	1.62297	23.9886	192.515	0.7991	29	31

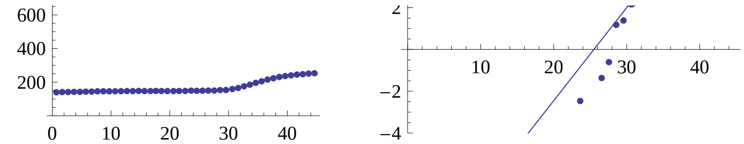
```
Export["C:\FilePath\FileName.xls", ReportResults];
```

```
(* If the plots below are too small,
grab the lower righthand corner of the array and pull it. *)
```

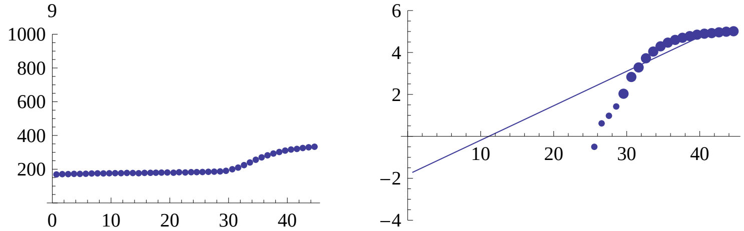
```
Show[GraphicsArray[ReportPlots]]
```



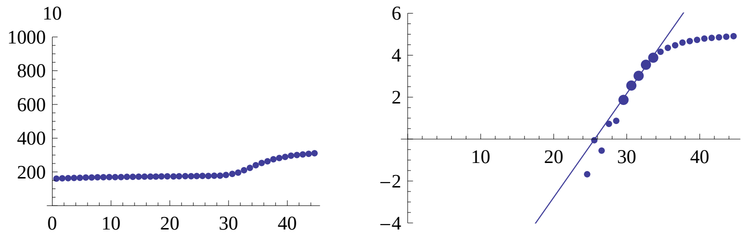

3
4
5
6
7
8



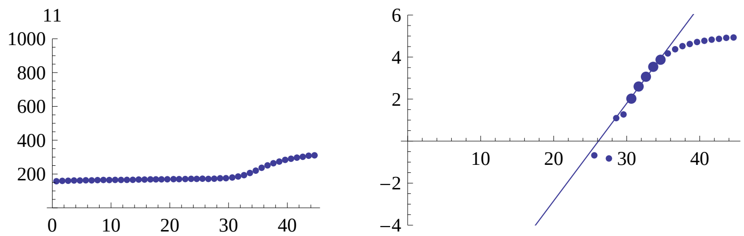
9



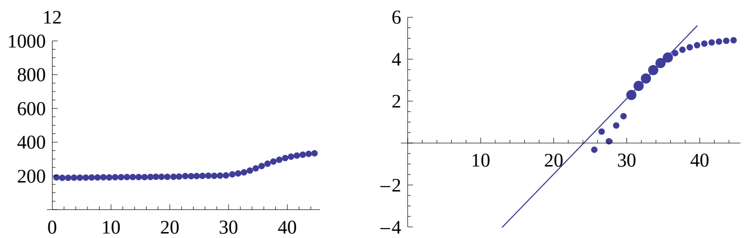
10



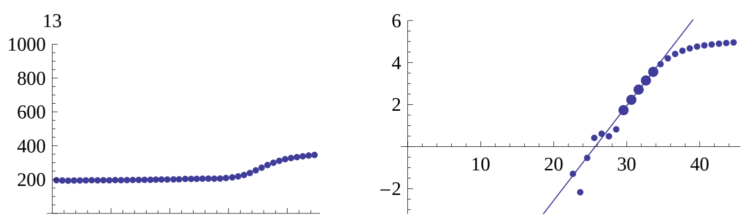
11



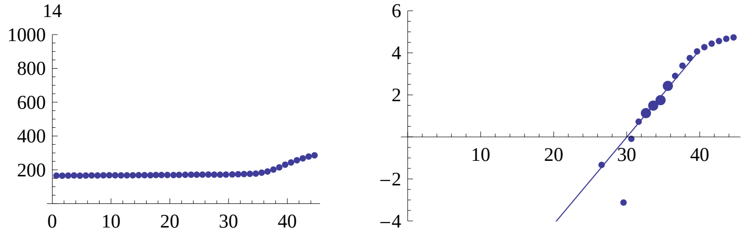
12



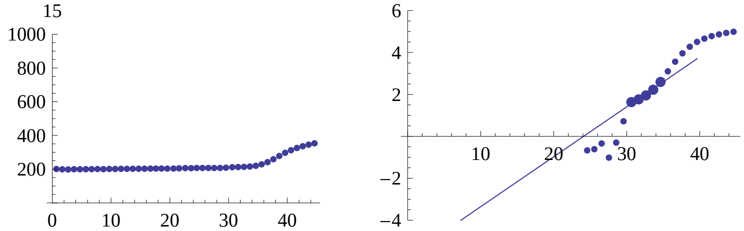
13



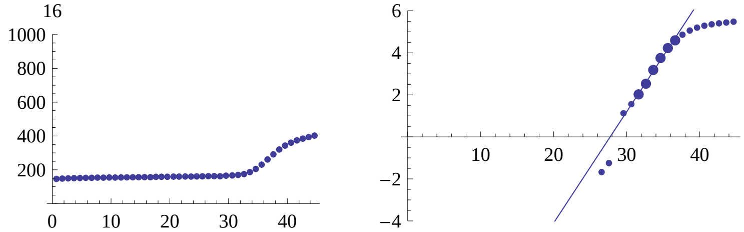
14



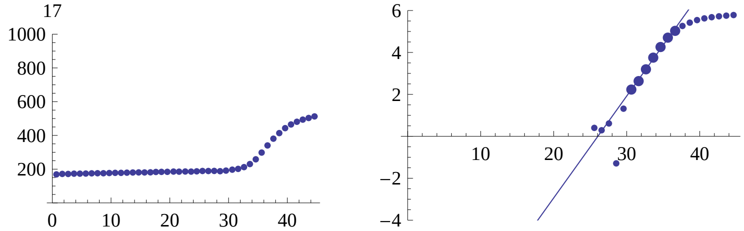
15



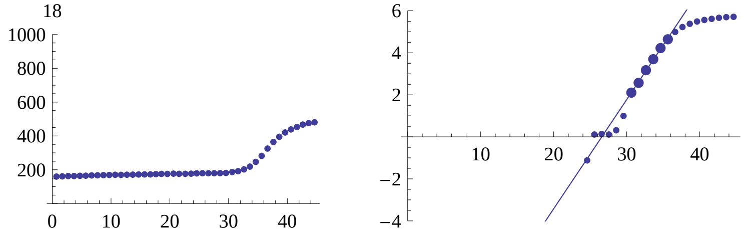
16



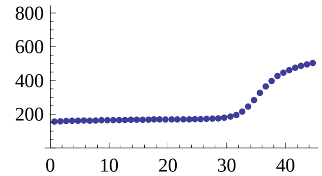
17

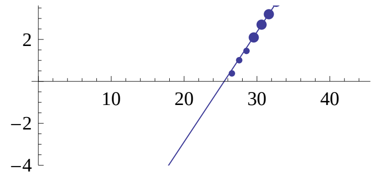


18

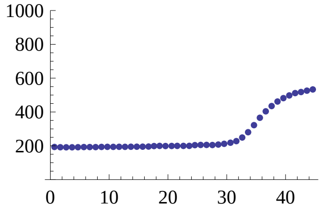


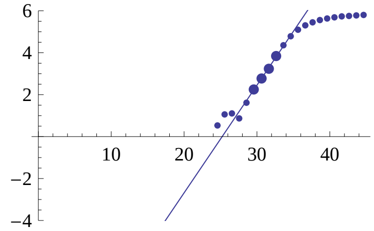
19


800
600
400
200
0
10
20
30
40

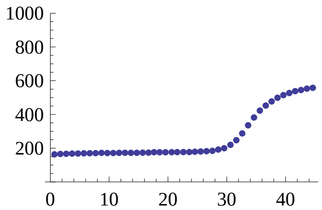

2
10
20
30
40
−2
−4

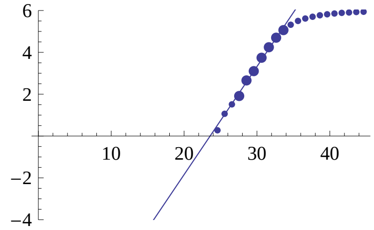
20


1000
800
600
400
200
0
10
20
30
40

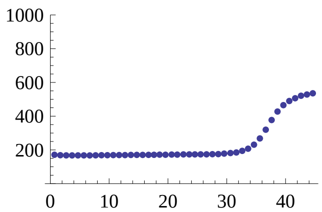

6
4
2
10
20
30
40
−2
−4

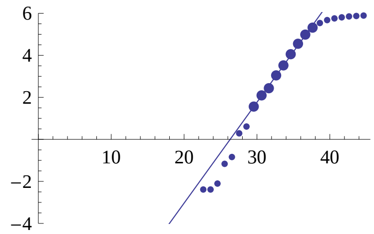
21


1000
800
600
400
200
0
10
20
30
40

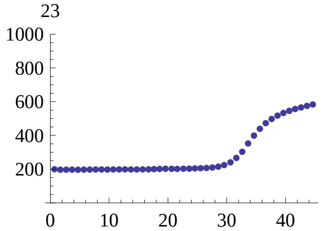

6
4
2
10
20
30
40
−2
−4

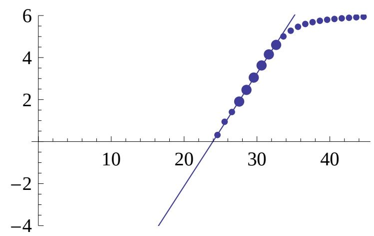
22


1000
800
600
400
200
0
10
20
30
40

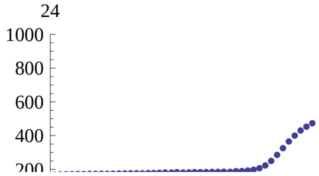

6
4
2
10
20
30
40
−2
−4

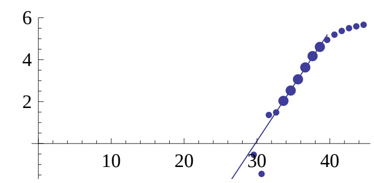
23

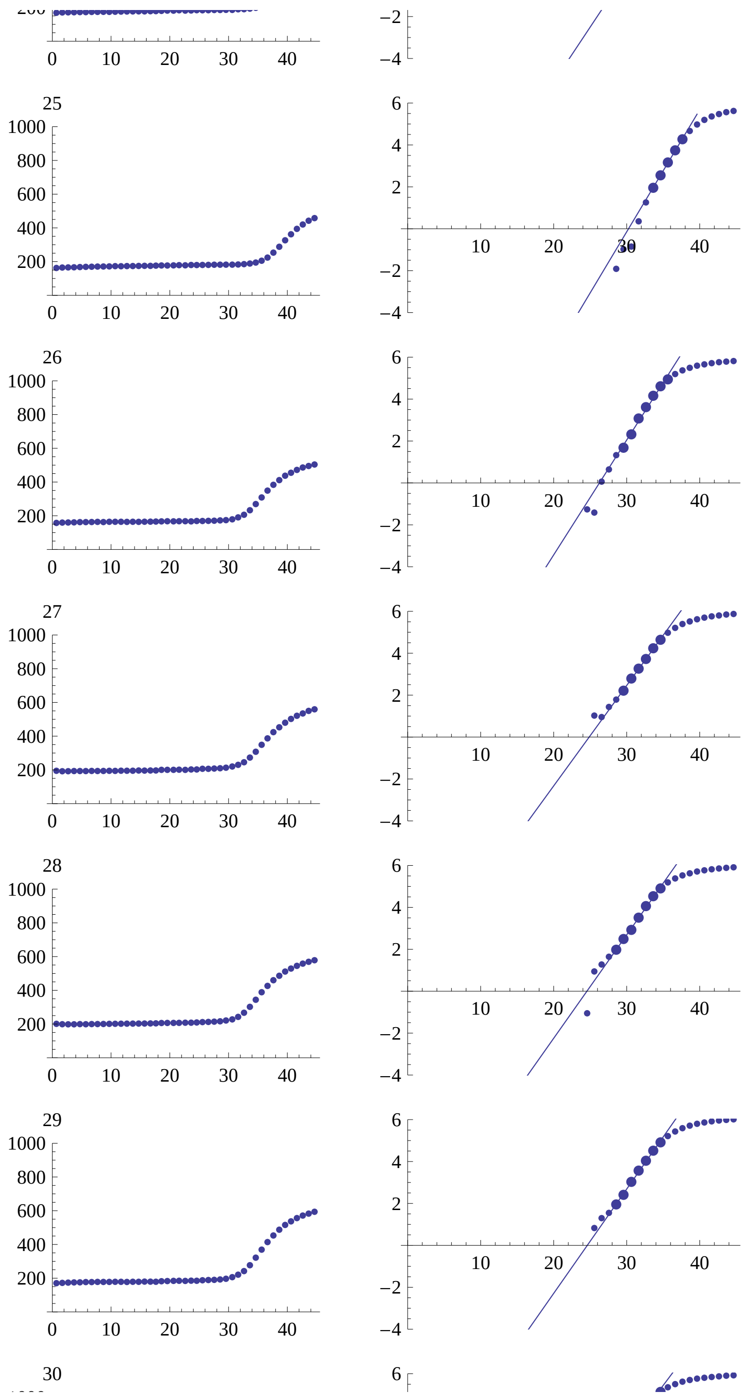

1000
800
600
400
200
0
10
20
30
40

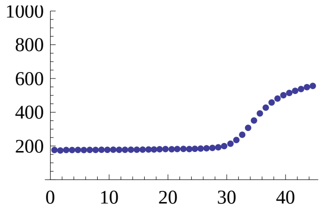

6
4
2
10
20
30
40
−2
−4

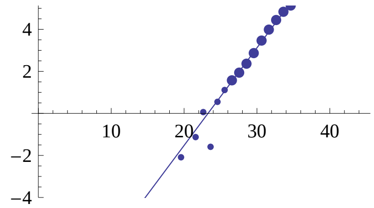
24


1000
800
600
400

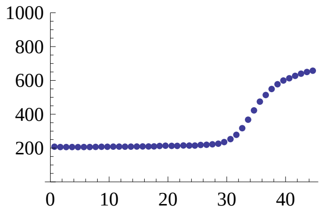

6
4
2
10
20
30
40

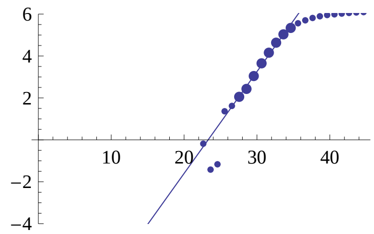

25
26
27
28
29
30


1000
800
600
400
200
0
10
20
30
40

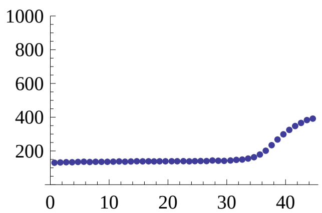

4
2
−2
−4
10
20
30
40

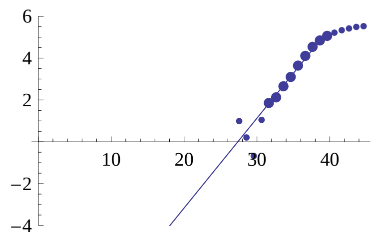
31


1000
800
600
400
200
0
10
20
30
40

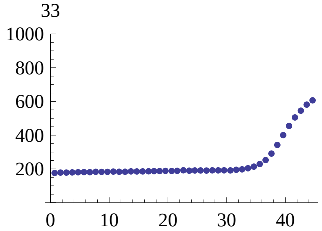

6
4
2
−2
−4
10
20
30
40

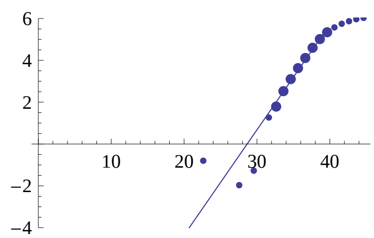
32


1000
800
600
400
200
0
10
20
30
40

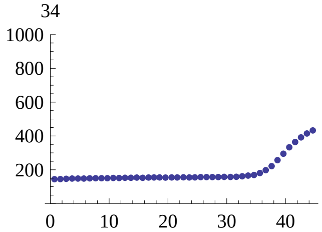

6
4
2
−2
−4
10
20
30
40

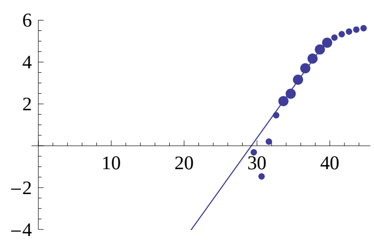
33


1000
800
600
400
200
0
10
20
30
40


6
4
2
−2
−4
10
20
30
40

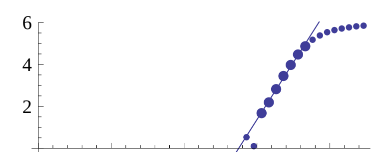
34

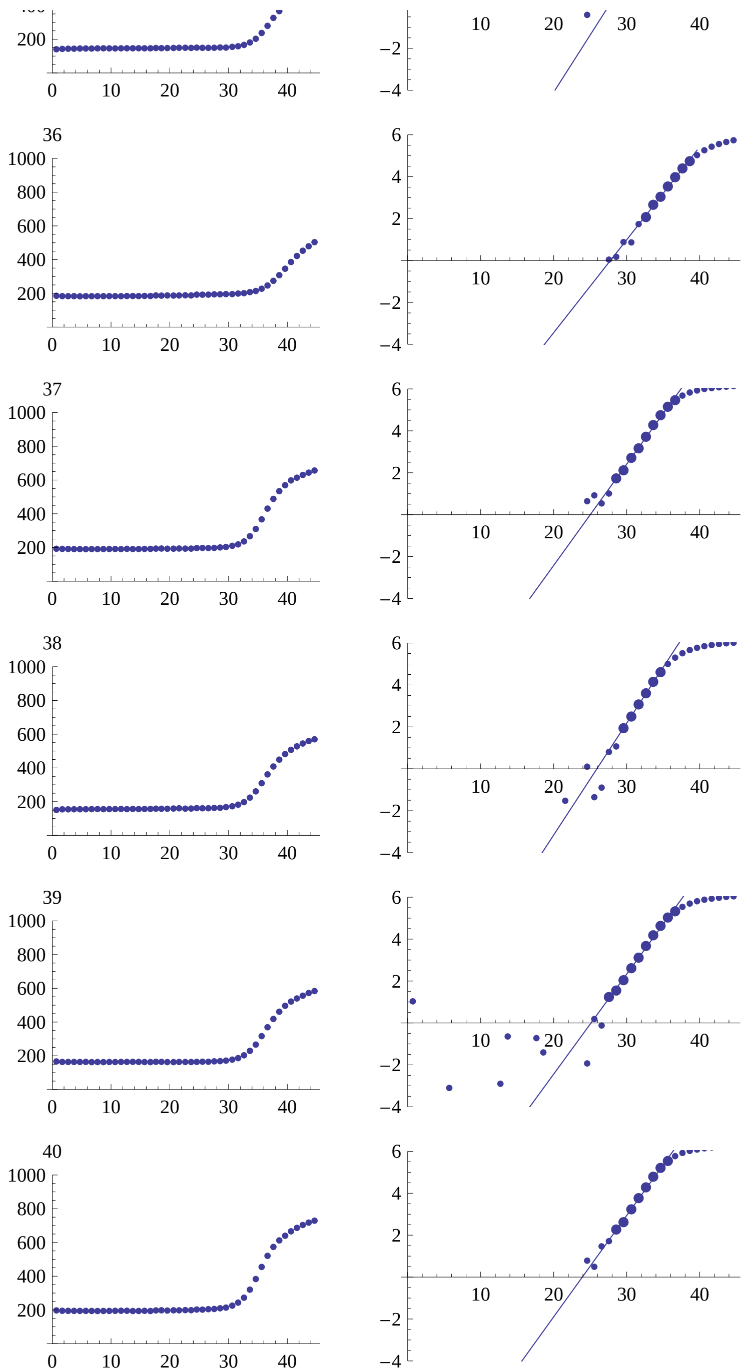

1000
800
600
400
200
0
10
20
30
40

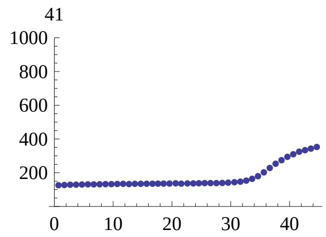
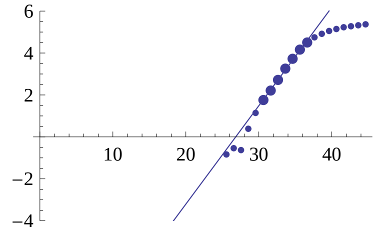

6
4
2
−2
−4
10
20
30
40

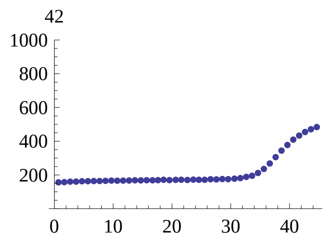
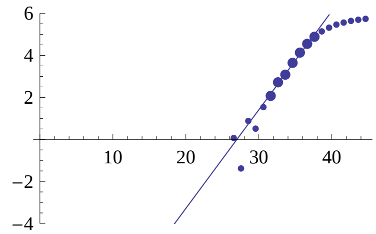
35

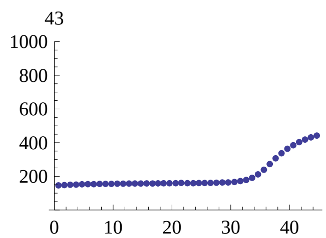
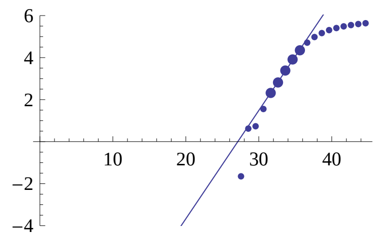

1000
800
600
400

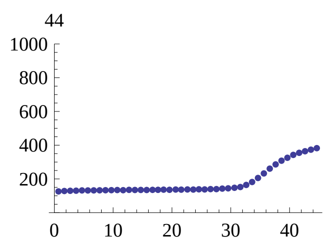
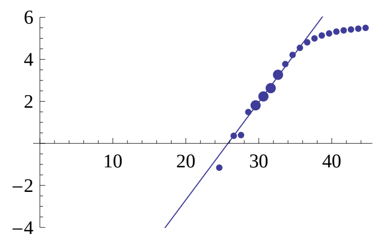

6
4
2

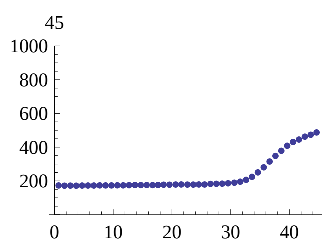
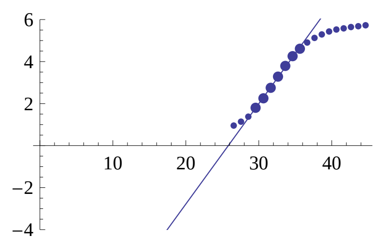

36
37
38
39
40


41


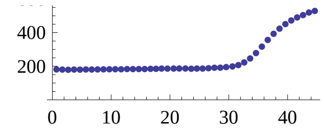

42


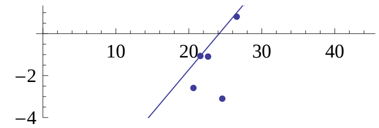

43



44


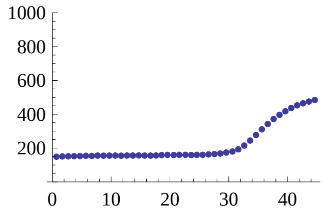

45


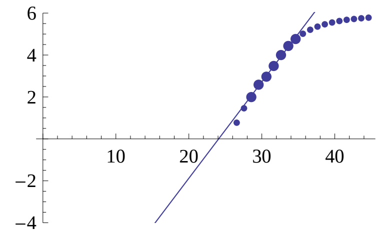

46



400
200
0
10
20
30
40

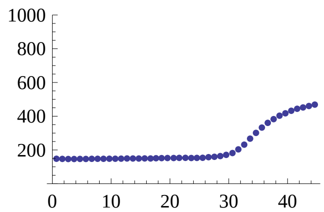

10
20
30
40
−2
−4

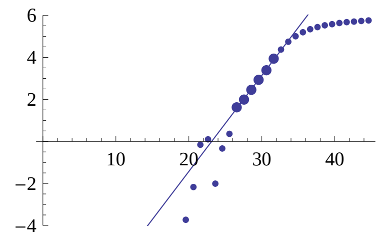
47


1000
800
600
400
200
0
10
20
30
40

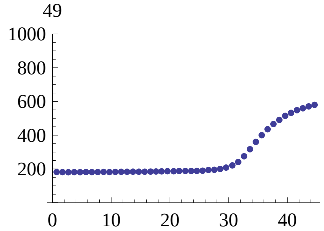

6
4
2
10
20
30
40
−2
−4

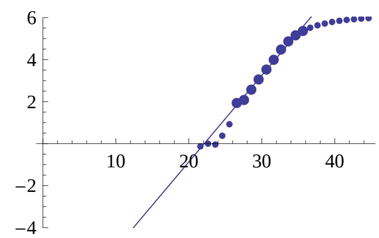
48


1000
800
600
400
200
0
10
20
30
40

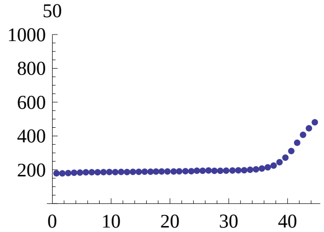

6
4
2
10
20
30
40
−2
−4

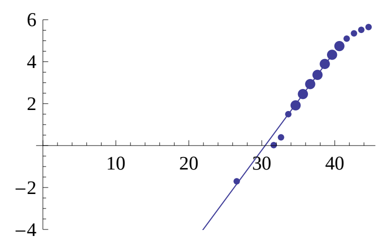
49


1000
800
600
400
200
0
10
20
30
40

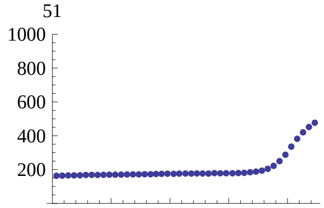

6
4
2
10
20
30
40
−2
−4

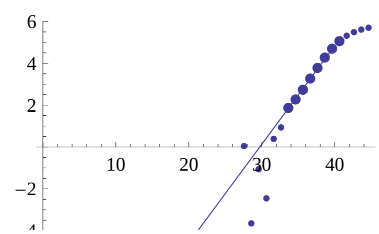
50

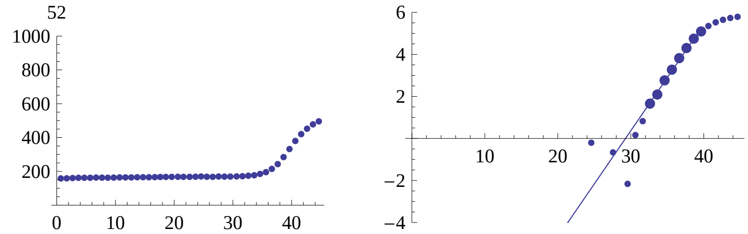

1000
800
600
400
200
0
10
20
30
40

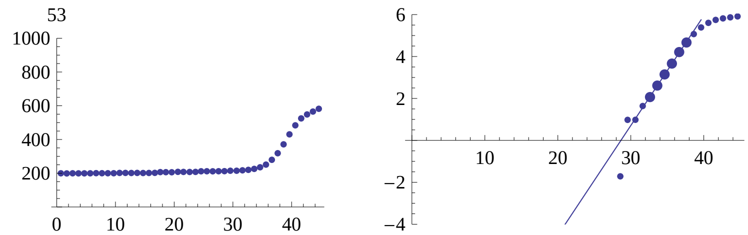

6
4
2
10
20
30
40
−2
−4

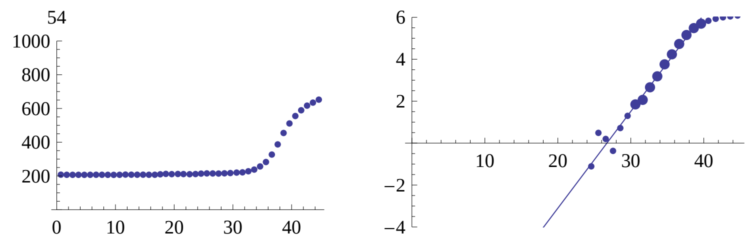
51

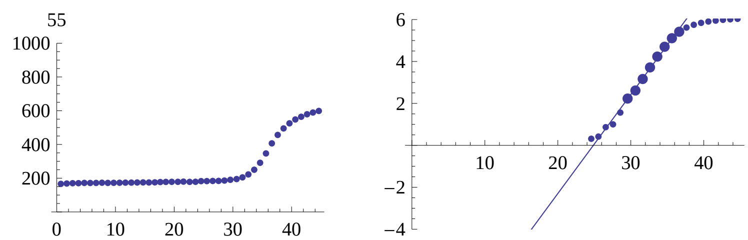

1000
800
600
400
200

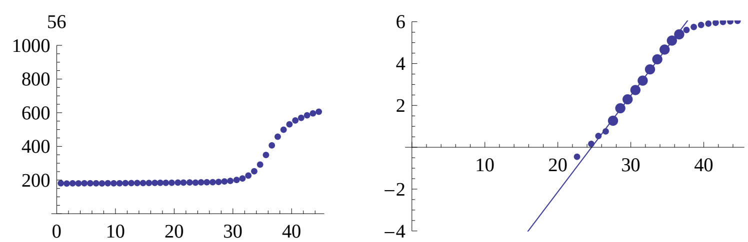

6
4
2
10
20
30
40
−2

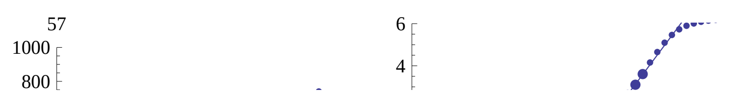

52

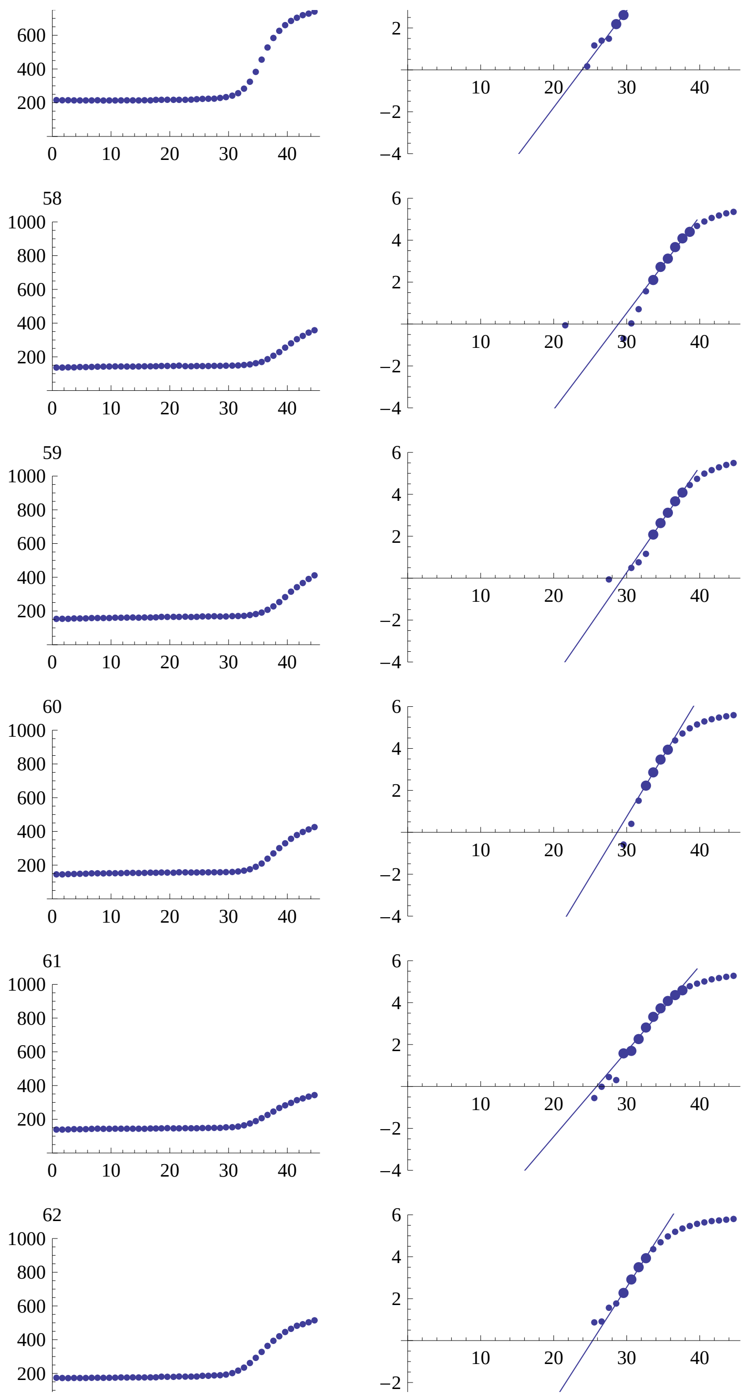

53

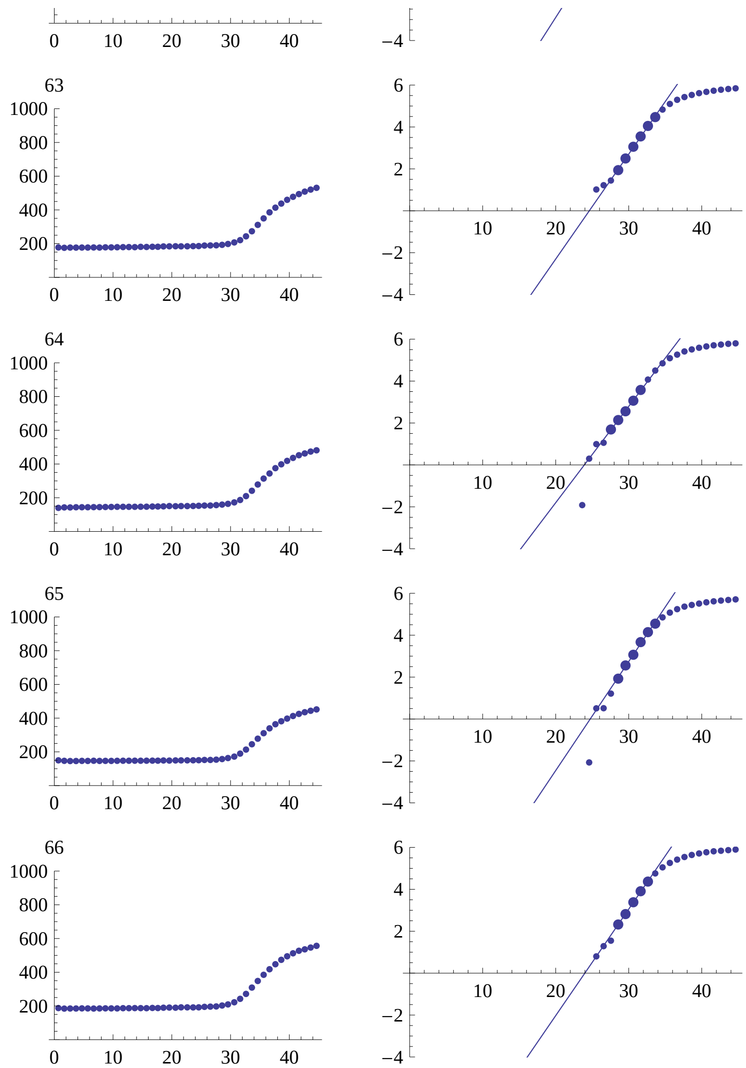

54


55


56


57

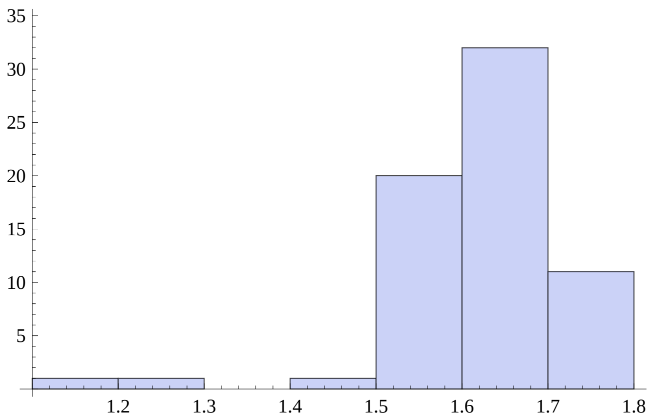

58
59
60
61
62




```
(* Examine the plots above and remove the amplifications where the slope was clearly not
    detected correctly or where a very small number of points was used to determine
    the slope. The number of such "failed" amplifications typically does not exceed
    10% and can be further improved by adjusting SigPStart, SigPEnd, SigPStartN,
   and SigPEndN. Here, such amplifications are #8, #9, #12, #14, and #15). Note
 that objective criteria (e.g., the 95% CI of the mean efficiency,
  numerical data from ReportResults, etc.) can be used to
 remove outliers. This functionality is currently not automated. *)

EfficiencySet = Drop[Transpose[ReportResults][[3]], 1];
```

```
Histogram[EfficiencySet]
```




```
MeanEfficiency = Mean[EfficiencySet]
```

1.62297

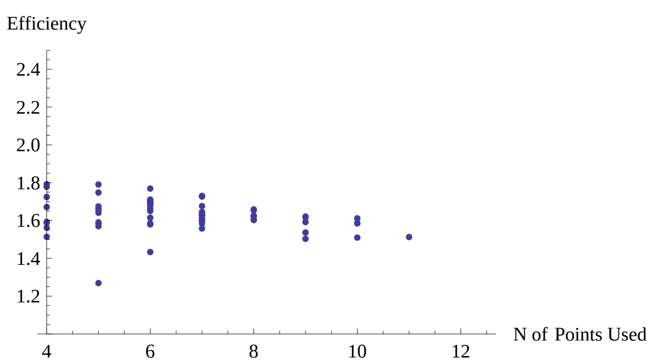
```
SDEfficiency = StandardDeviation[EfficiencySet]
```

0.100937

```
EfficiencyPoints = Length[EfficiencySet]
```

66

```
EfficiencyPointNumbers = Drop[Transpose[ReportResults][[11]], 1];
EfficiencyEstimates = Drop[Transpose[ReportResults][[3]], 1];
ListPlot[MapThread[List, {EfficiencyPointNumbers, EfficiencyEstimates}],
 PlotRange → {1.0, 2.5}, AxesLabel → {"N of Points Used", "Efficiency"}]
```




```
(* Use only if outliers are present and have to be removed. *)
```

```
(* Set OutlierQ=1 if outliers are present. *)
OutlierQ = 0;

(* List the outlier numbers *)
OutlierIDs = {0};

If[OutlierQ == 1,
 NewEfficiencySet = Drop[EfficiencySet, OutlierIDs];
 Print["New Mean = ", Mean[NewEfficiencySet]];
 Print["New SD = ", StandardDeviation[NewEfficiencySet]];
 Print["New Points = ", Length[NewEfficiencySet]]
]
```