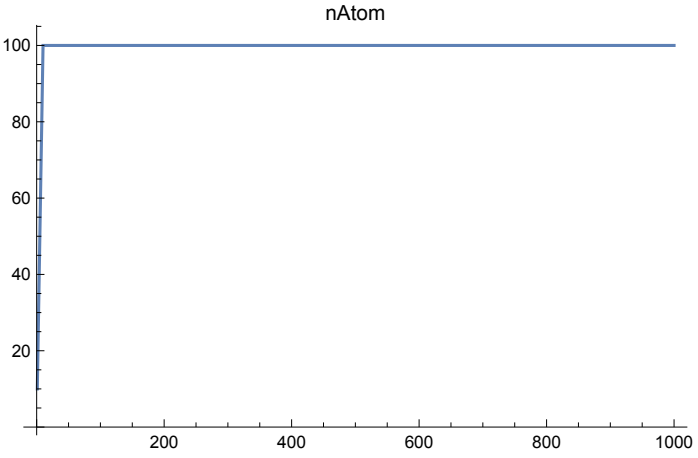
In[359]:=
```
Clear["Global`*"]
```

In[360]:=
```
inputData = Import[NotebookDirectory[] <> "/input.txt"];
input = StringSplit[inputData];
l = Length[input];(*l will have to be even*)
(*Write a function here that will take in strings as variable names*)
dt = Read[StringToStream[input[[2]]]];
tmax = input[[4]];
nTrajectory = input[[6]];
nstore = input[[8]];
yWall = input[[10]];
sigmaXX = input[[12]];
sigmaXZ = input[[14]];
transitTime = input[[16]];
sigmaPX = input[[18]];
sigmaPY = input[[20]];
sigmaPZ = input[[22]];
density = input[[24]];
rabi = input[[26]];
kappa = input[[28]];
lambda = input[[30]];
invT2 = input[[32]];
name = input[[34]]
```

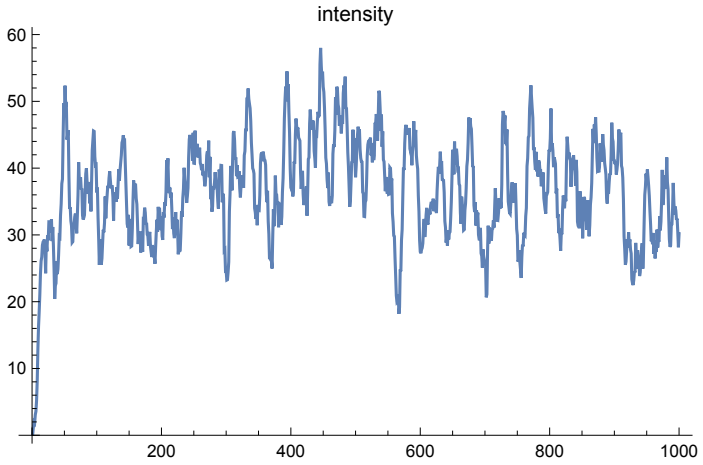
Out[379]=
```
test2
```

In[380]:=
```
(*When doing one run*)
intensity = Flatten[Import[NotebookDirectory[] <> name <> "/intensity.dat"]];
nAtom = Flatten[Import[NotebookDirectory[] <> name <> "/nAtom.dat"]];
inversionAve =
  Flatten[Import[NotebookDirectory[] <> name <> "/inversionAve.dat"]];
spinSpinCorAveRe = Flatten[
   Import[NotebookDirectory[] <> name <> "/spinSpinCorAve_re.dat"]];
spinSpinCorAveIm = Flatten[Import[
    NotebookDirectory[] <> name <> "/spinSpinCorAve_im.dat"]];
szFinal = Flatten[Import[NotebookDirectory[] <> name <> "/szFinal.dat"]];
realG1 = Import[NotebookDirectory[] <> name <> "/realG1.dat"];
spectra = Import[NotebookDirectory[] <> name <> "/spectra.dat"];
ListLinePlot[nAtom, PlotRange → All, PlotLabel → "nAtom"]
ListLinePlot[intensity, PlotRange → All, PlotLabel → "intensity"]
ListLinePlot[inversionAve, PlotRange → All, PlotLabel → "inversionAve"]
ListLinePlot[spinSpinCorAveRe, PlotRange → All, PlotLabel → "spinSpinCorAve_re"]
ListLinePlot[spinSpinCorAveIm, PlotRange → All, PlotLabel → "spinSpinCorAve_im"]
ListPlot[szFinal, PlotRange → All, PlotLabel → "inversionOut"]
ListLinePlot[realG1, PlotRange → All, PlotLabel → "realG1"]
ListLinePlot[spectra, PlotRange → All, PlotLabel → "spectra"]
```

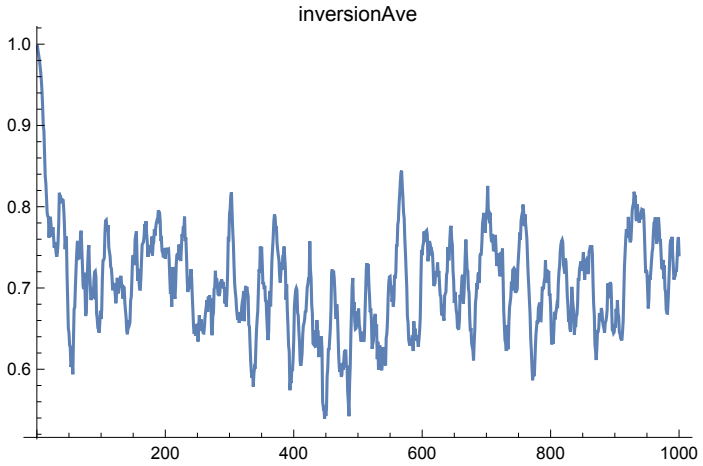
Out[388]=



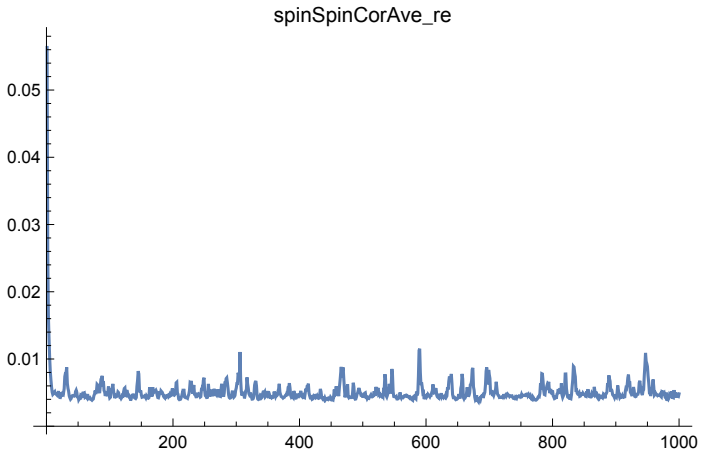
Out[389]=



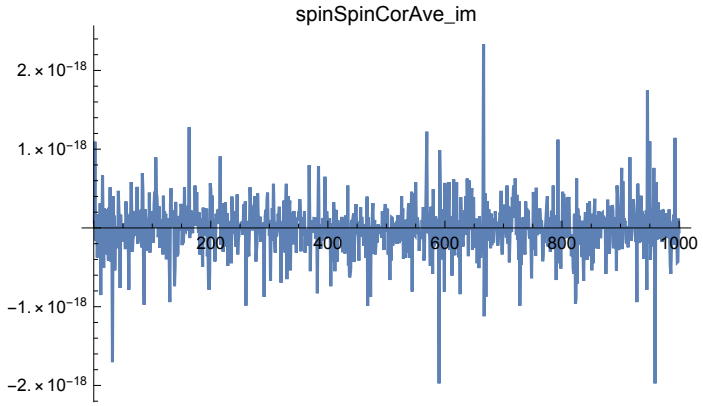
Out[390]=



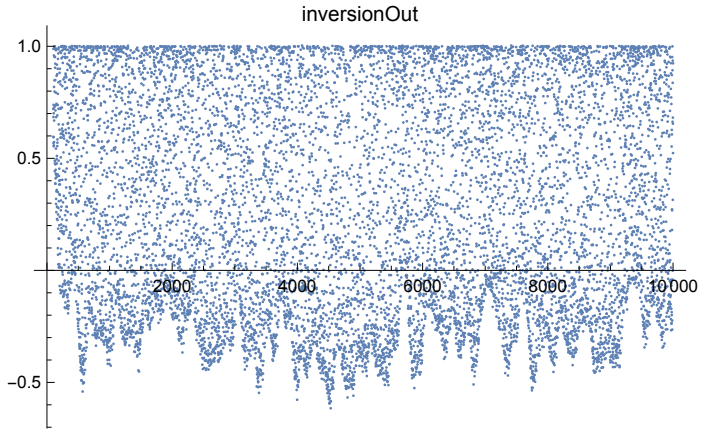
Out[391]=



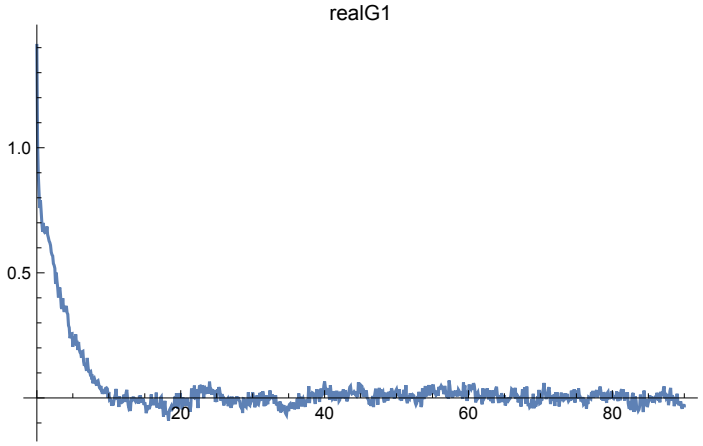
Out[392]=



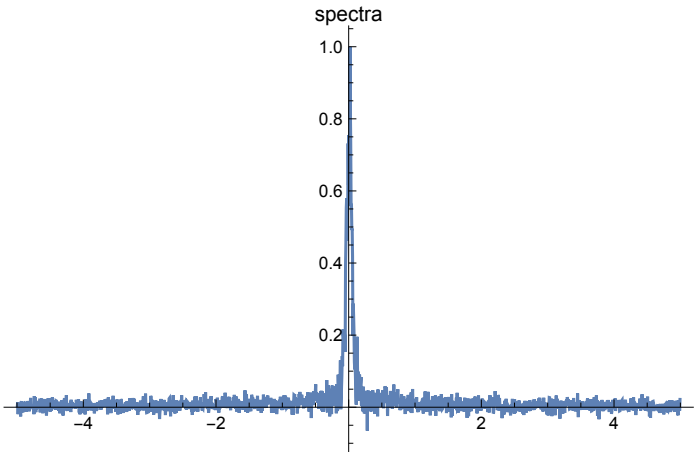
Out[393]=



Out[394]=



Out[395]=



In[396]:=
```
(*Fit the spectra to lorentzian*)
lorentzianModel = A linewidth/(linewidth^2 + 4 x^2);
fitLoren = NonlinearModelFit[spectra, {lorentzianModel}, {linewidth, A}, {x}]
fitLoren["ParameterTable"]
linewidth1 = linewidth /. fitLoren["BestFitParameters"]
Show[
 {Plot[fitLoren[x], {x, -0.3, 0.3}, PlotRange → All, BaseStyle → {FontSize → 30}],
  Graphics[{Red, PointSize[.01], Map[Point, spectra]}]},
 Frame → True, FrameLabel → {"v/Transit time^-1", "Power Spectra"}]
```

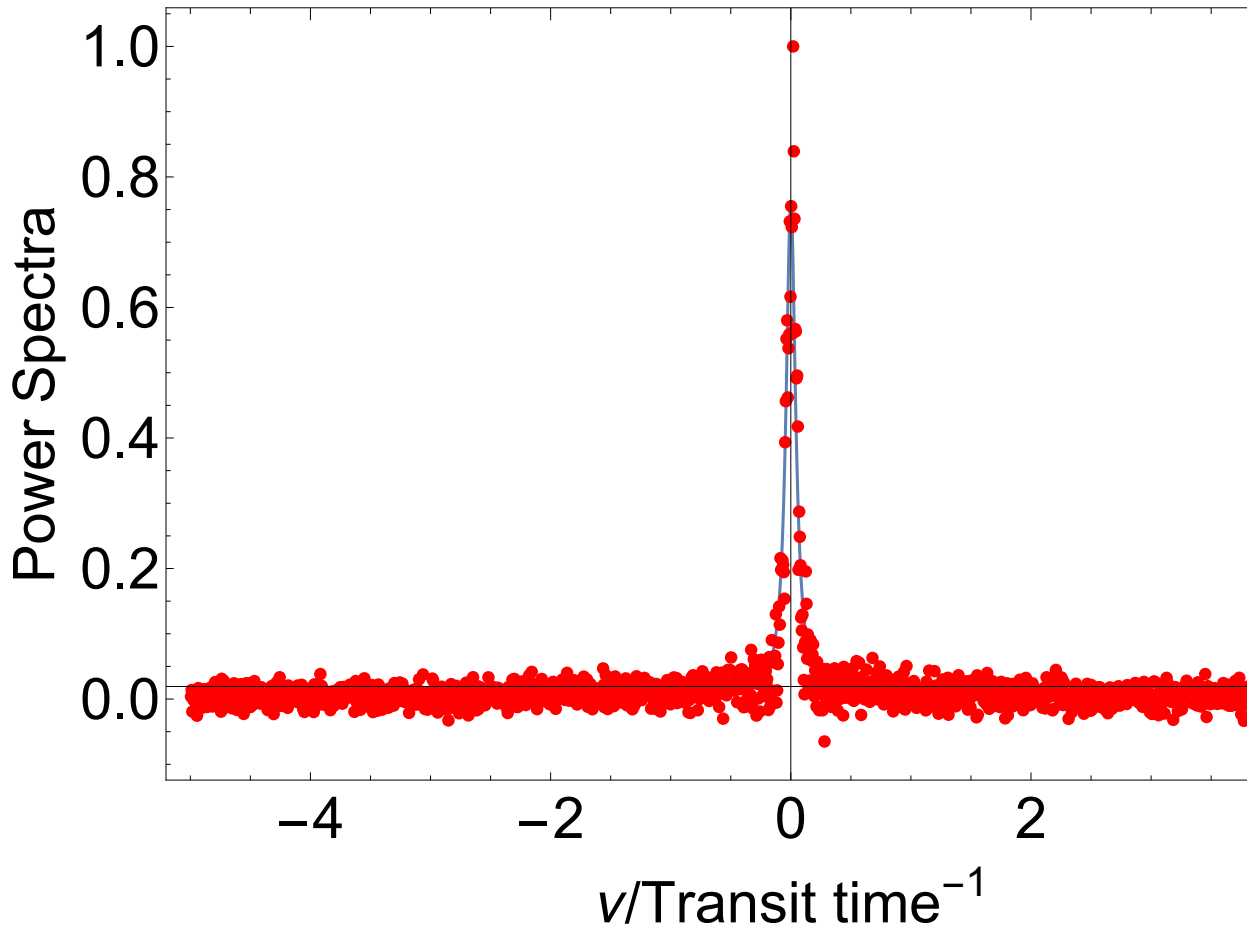
Out[397]=
$$\text{FittedModel}\left[\frac{0.00711117}{0.00932305 + 4\,x^2}\right]$$

Out[398]=

	Estimate	Standard Error	t-Statistic	P-Value
linewidth	0.0965559	0.0015798	61.1191	$2.85400345851 \times 10^{-441}$
A	0.0736482	0.00085206	86.4355	$1.024768773445 \times 10^{-642}$

Out[399]= 0.0965559

Out[400]=



In[401]:=
$$\text{tauc1} = \frac{1}{\text{linewidth1 Pi}}$$

Out[401]= 3.29664

In[402]:=
```
(*Fit the G1 function to exponential*)
exponentialModel = B Exp[- t / tc]
fitExp = NonlinearModelFit[realG1, {exponentialModel}, {tc, B}, {t}]
fitExp["ParameterTable"]
tauc2 = tc /. fitExp["BestFitParameters"]
Show[{Plot[fitExp[x], {x, 0, Last[realG1][[1]]}, PlotRange → All,
    PlotStyle → {Thickness[0.01]}, BaseStyle → {FontSize → 30}],
  Graphics[{Red, PointSize[.001], Map[Point, realG1]}]},
 Frame → True, FrameLabel → {"Time/T", "G1Slow"}]
```

Out[402]= $B\, e^{-\frac{t}{tc}}$

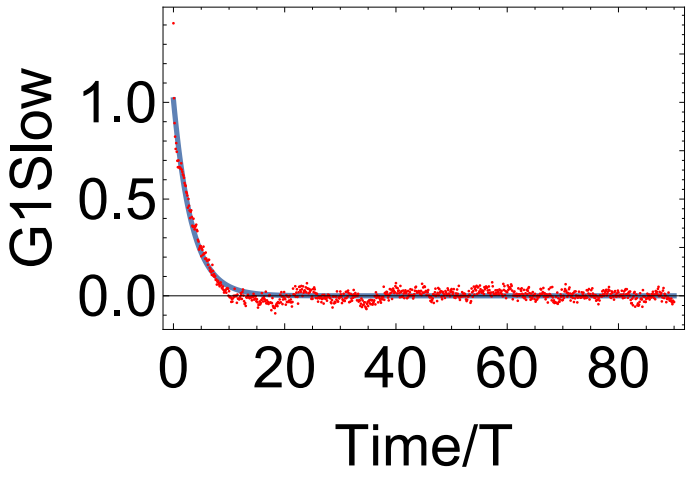
Out[403]= FittedModel[$1.01307\, e^{-0.303048\, t}$]

Out[404]=

	Estimate	Standard Error	t-Statistic	P-Value
tc	3.29981	0.051147	64.5163	$1.301604028190 \times 10^{-339}$
B	1.01307	0.0109373	92.6258	$4.13454035930 \times 10^{-462}$

Out[405]= 3.29981

Out[406]=



In[407]:=
```
(*Get the linewidth from the coherent time*)
linewidth2 = 1 / (tauc2 Pi)
```

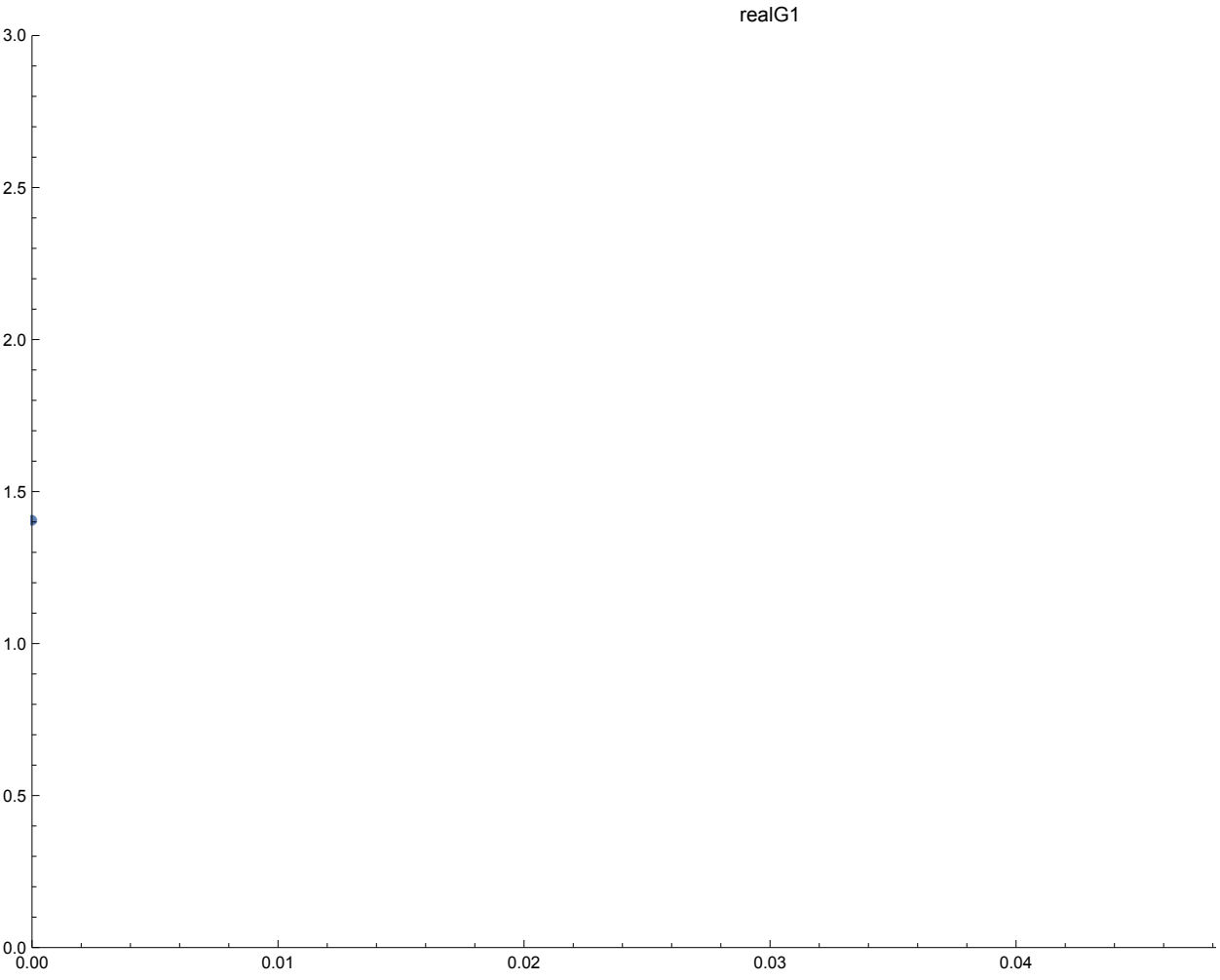
Out[407]= 0.096463

In[408]:=

In[409]:=
```
(*Notice there is another exponential decay.*)
```

In[410]:=
```
cutOff = 0.06;
ListPlot[realG1, PlotRange → {{0, cutOff}, {0, 3}},
 PlotLabel → "realG1", PlotMarkers → {Automatic, Small}]
secondNum = IntegerPart[cutOff / realG1[[2, 1]]];
secondExp = Take[realG1, secondNum];
exponentialModel2 = B2 Exp[- Pi linewidthQuick t]
fitExp2 =
 NonlinearModelFit[secondExp, {exponentialModel2}, {linewidthQuick, B2}, {t}]
fitExp2["ParameterTable"]
Show[{Plot[fitExp2[x], {x, 0, cutOff}, PlotRange → All, BaseStyle → {FontSize → 30}],
  Graphics[{Red, PointSize[.02], Map[Point, secondExp]}]},
 Frame → True, FrameLabel → {"Time/T", "G1Fast"}]
```

Out[411]=



Out[414]= $B2\, e^{-\text{linewidthQuick}\,\pi\, t}$

NonlinearModelFit: First argument {} in NonlinearModelFit is not a list or a rectangular array.

Out[415]= $\text{NonlinearModelFit}\left[\{\}, \left\{B2\, e^{-\text{linewidthQuick}\,\pi\, t}\right\}, \{\text{linewidthQuick}, B2\}, \{t\}\right]$

Out[416]= NonlinearModelFit[{}, {B2 e^(-linewidthQuick π t)}, {linewidthQuick, B2}, {t}][ParameterTable]

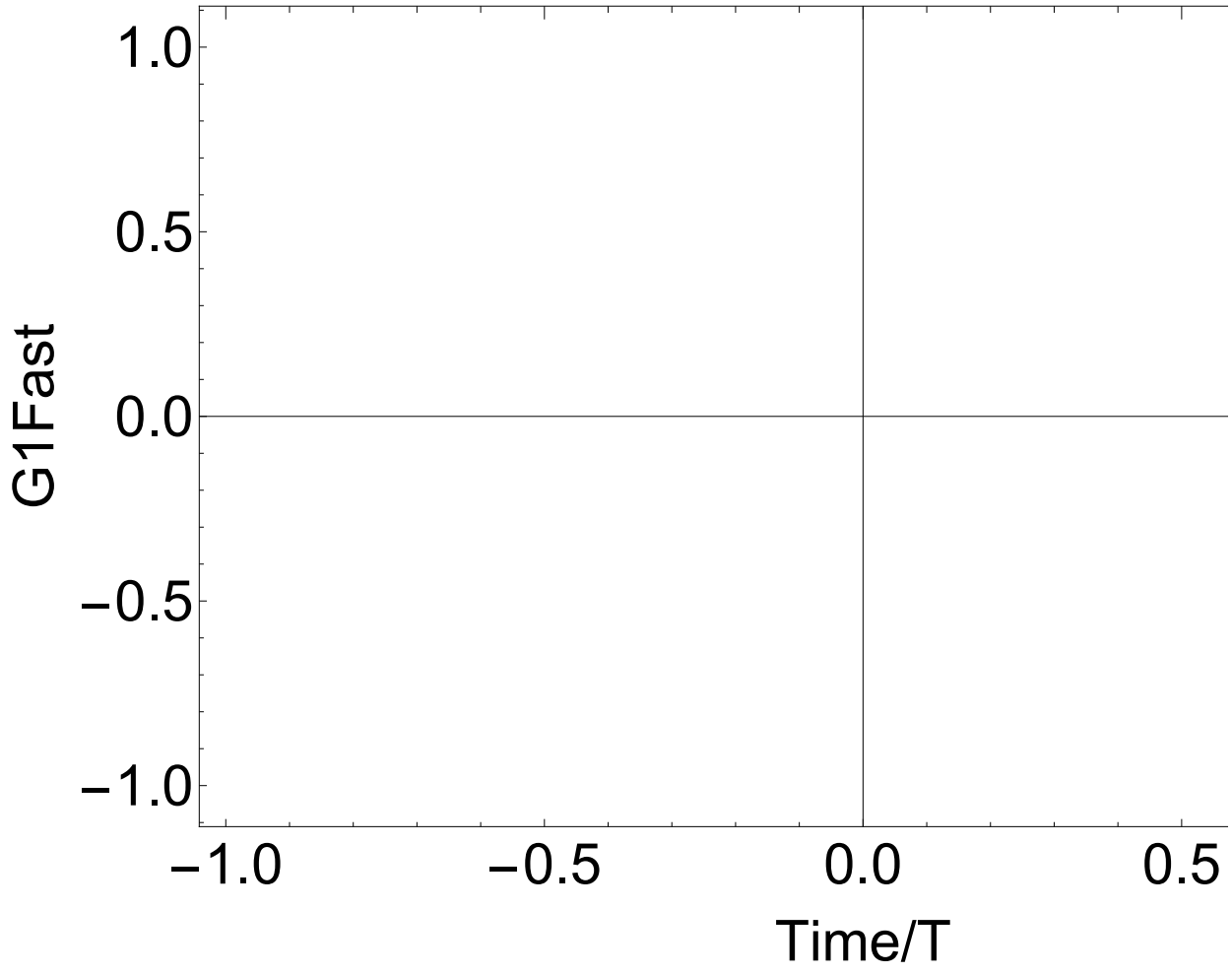
... NonlinearModelFit: First argument {} in NonlinearModelFit is not a list or a rectangular array.

... NonlinearModelFit: First argument {} in NonlinearModelFit is not a list or a rectangular array.

... NonlinearModelFit: First argument {} in NonlinearModelFit is not a list or a rectangular array.

... General: Further output of NonlinearModelFit::fitd will be suppressed during this calculation.

Out[417]=



In[418]:=
```
secondTc = 1/(linewidthQuick Pi) /. fitExp2["BestFitParameters"]
```

... ReplaceAll: {NonlinearModelFit[{}, {B2 e^(-linewidthQuick π t)}, {linewidthQuick, B2}, {t}][BestFitParameters]} is neither a list of replacement rules nor a valid dispatch table, and so cannot be used for replacing.

Out[418]=
```
1/(linewidthQuick π) /.
 NonlinearModelFit[{}, {B2 e^(-linewidthQuick π t)}, {linewidthQuick, B2}, {t}][
  BestFitParameters]
```