Chapter 4 pt. 1: Energy efficient QND measurementand read-out

Author: Bruno Ortega Goes

This code organized written by me ensembles in a organized manner my own codes and some codes produced by Stephen Wein, PhD. It produces the images presented in the paper arXiv:2205.09623 and in the thesis.

```
(*Setting the path where the plots are going to be saved.*)
nbdirectory = SetDirectory[NotebookDirectory[]];
plotsPath = nbdirectory <> "/PlotsChap4/";

Clear[savePlot];
savePlot[nameAndExtension_, plot_, path_ : plotsPath] :=
 If[Length@path == 0, Export[path <> nameAndExtension, plot], Do[
   Export[PlotsPath <> nameAndExtension, plot], {PlotsPath, path}]]

fontsize = 24;
```

Vectorization☑

Hilbert Space ☑

In[17]:=
```
dim4LS = 4;
dimSource = 4;
dim = dim4LS dimSource^2;
(*Ordering the 4LS basis as:*)
{spinDw, spinUp, trionDw, trionUp} =
  {{1, 0, 0, 0}, {0, 1, 0, 0}, {0, 0, 1, 0}, {0, 0, 0, 1}};
```

In[21]:=
```
(*Dipole operators*)
(*This is enough for the semiclassical model*)
ΣL = Outer[Times, spinDw, trionDw];
ΣR = Outer[Times, spinUp, trionUp];
```
$$\Sigma H = \frac{\Sigma R + \Sigma L}{\sqrt{2}};$$
$$\Sigma V = \frac{\Sigma R - \Sigma L}{i\sqrt{2}};$$
$$\Sigma D = \frac{\Sigma H + \Sigma V}{\sqrt{2}};$$
$$\Sigma A = \frac{\Sigma H - \Sigma V}{\sqrt{2}};$$

In[27]:=
```
(*Dipole Operators in the large Hilbert Space*)
(*This is necessary for the SLH framework*)
σL = kp[ΣL, eye[dimSource], eye[dimSource]];
σR = kp[ΣR, eye[dimSource], eye[dimSource]];
```

In[29]:=
```
(*Virtual cavity operators*)
aR = kp[eye[dim4LS], a[dimSource], eye[dimSource]] // N;
aL = kp[eye[dim4LS], eye[dimSource], a[dimSource]] // N;
```

Pre-measurement: qBhat ☑

Semiclassical drive ☑

Defining the physical system:

In[50]:=
```
Hsemiclassical = i α (ΣH - ΣH†) // mf;
ℒsemiclassical = unitary2vec[Hsemiclassical] +
      γ lindblad2vec[ΣL] + γ lindblad2vec[ΣR] // Refine // mf;
dimSC = Length@Hsemiclassical;
traceSC = Flatten[eye[dimSC]];
```

$$\begin{pmatrix} 0 & 0 & \frac{i\alpha}{\sqrt{2}} & 0 \\ 0 & 0 & 0 & \frac{i\alpha}{\sqrt{2}} \\ -\frac{i\alpha}{\sqrt{2}} & 0 & 0 & 0 \\ 0 & -\frac{i\alpha}{\sqrt{2}} & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix}
0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 & \gamma & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 \\
-\frac{\alpha}{\sqrt{2}} & 0 & -\frac{\gamma}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{\alpha}{\sqrt{2}} & 0 & -\frac{\gamma}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 & \gamma \\
0 & 0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & -\frac{\gamma}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} & 0 \\
0 & 0 & 0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & -\frac{\gamma}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{\alpha}{\sqrt{2}} \\
-\frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{\gamma}{2} & 0 & \frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 \\
0 & -\frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{\gamma}{2} & 0 & \frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 \\
0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & -\gamma & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & -\gamma & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{\gamma}{2} & 0 & \frac{\alpha}{\sqrt{2}} & 0 \\
0 & 0 & 0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{\gamma}{2} & 0 & \frac{\alpha}{\sqrt{2}} \\
0 & 0 & 0 & 0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & -\gamma & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & 0 & 0 & 0 & 0 & -\frac{\alpha}{\sqrt{2}} & 0 & -\gamma
\end{pmatrix}$$

Initial state and qBhat computer:

In[54]:=
```
ψ04LS = (spinUp + spinDw)/√2;
vecρ04LS = Flatten[kp[ψ04LS, ψ04LS]];
```

In[56]:=
```
σupdw = Outer[Times, spinDw, spinUp] // mf;
σupdw = leftacting[σupdw] // mf;
```

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix}
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

Finding the steady state qBhat, 𝓑qcsSS (this is an efficient method based on Dr. Wein’s code):

In[58]:=
```
𝓑qcsSS = Limit[
  (traceSC.σupdw.MatrixExp[ℒsemiclassical t /. α → 0].MatrixExp[ℒsemiclassical τ,
      vecρ04LS]) /. γ → 1 /. τ → 1/Γ /. α → Sqrt[nH Γ], t → ∞]
```

Out[58]=

$$\frac{1}{4\times(1-8\,\mathrm{nH}\,\Gamma)}\,e^{-\frac{1}{2}/\Gamma}\left(e^{\frac{1}{2\Gamma}+\frac{-1-\sqrt{1-8\,\mathrm{nH}\,\Gamma}}{2\Gamma}}+e^{\frac{1}{2\Gamma}+\frac{-1+\sqrt{1-8\,\mathrm{nH}\,\Gamma}}{2\Gamma}}-8\,\mathrm{nH}\,\Gamma-4\,e^{\frac{1}{2\Gamma}+\frac{-1-\sqrt{1-8\,\mathrm{nH}\,\Gamma}}{2\Gamma}}\,\mathrm{nH}\,\Gamma-4\,e^{\frac{1}{2\Gamma}+\frac{-1+\sqrt{1-8\,\mathrm{nH}\,\Gamma}}{2\Gamma}}\,\mathrm{nH}\,\Gamma-e^{\frac{1}{2\Gamma}+\frac{-1-\sqrt{1-8\,\mathrm{nH}\,\Gamma}}{2\Gamma}}\sqrt{1-8\,\mathrm{nH}\,\Gamma}+e^{\frac{1}{2\Gamma}+\frac{-1+\sqrt{1-8\,\mathrm{nH}\,\Gamma}}{2\Gamma}}\sqrt{1-8\,\mathrm{nH}\,\Gamma}\right)$$

Here there is a 3.8 re-scaling of the Γ.

In[59]:=
```
coherentsetsemiclassical =
  N@ParallelTable[{10^x, 10^y/3.8, Chop[𝓑qcsSS /. {nH → 10^x, Γ → 10^y/3.8}]},
    {x, -1, 2, 2×10^-2}, {y, -2, 2, 2×10^-2}];
 (*The Γ is re-scaled by a factor 3.8*)
```

In[61]:=
```
coherentsetsemiclassical = Flatten[coherentsetsemiclassical, 1]
```

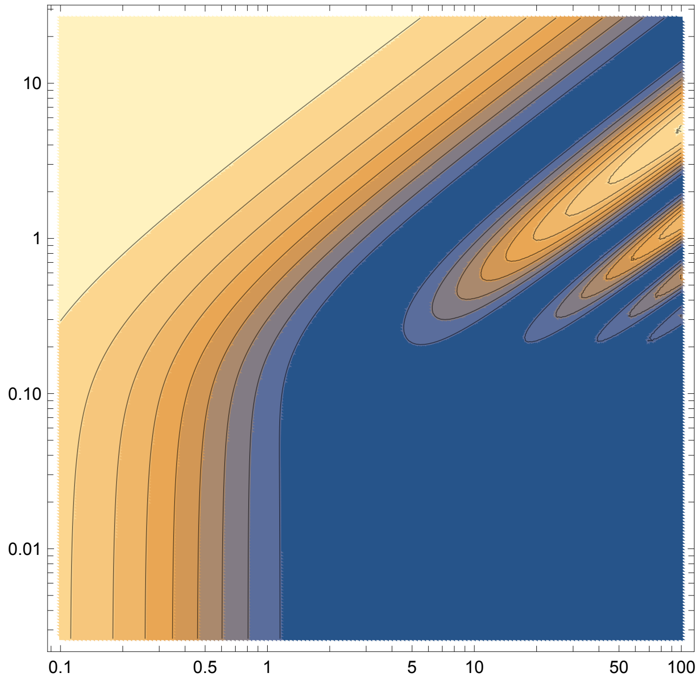
Out[61]=
```
{{0.1, 0.00263158, 0.409754}, {0.1, 0.0027556, 0.409772},
 … 30 348 …, {100., 26.3158, 0.0192338}}
```

large output | show less | show more | show all | set size limit...

First look at the plot:

In[62]:=
```
ListContourPlot[coherentsetsemiclassical, ScalingFunctions → {"Log", "Log"}]
```

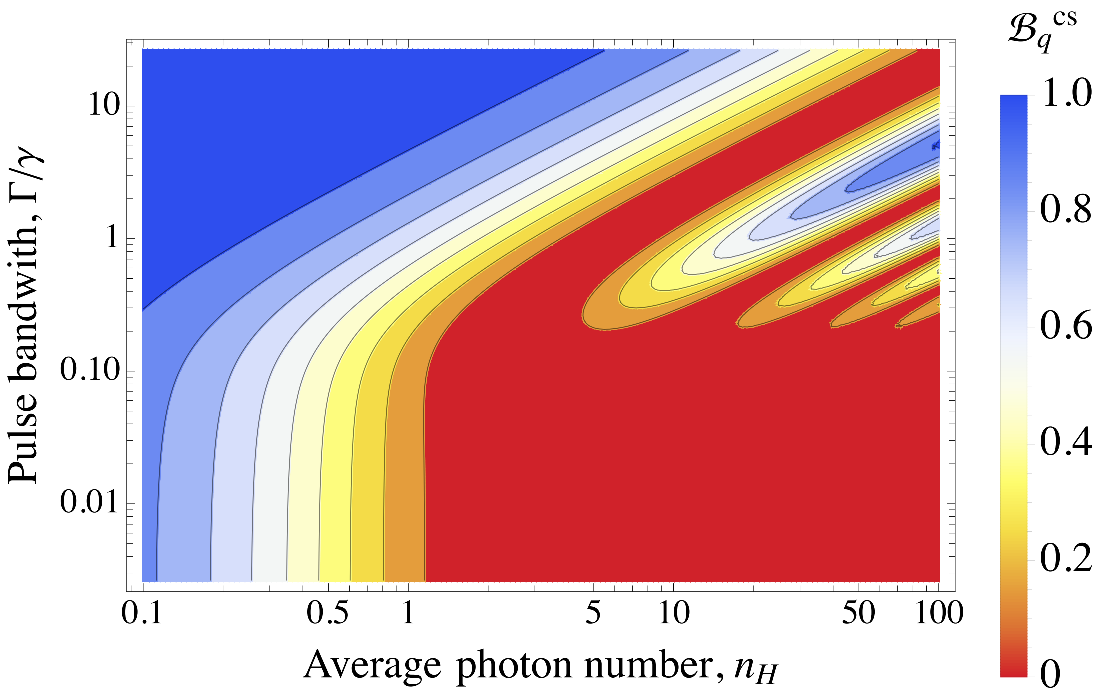
Out[62]=



Making it stylish:

In[75]:=
```
(*Define a custom color function*)
colorf = ColorData[{"TemperatureMap", "Reverse"}][Rescale[#, {0, 1}]] &;

Fig1 = ListContourPlot[coherentsetsemiclassical,
  AspectRatio → 10 / 15,
  ImageSize → 500,
  ScalingFunctions → {"Log", "Log"},
  ColorFunction → colorf,
  ColorFunctionScaling → True,
  FrameLabel → {Style["Average photon number, nH", 22, Black],
    Style["Pulse bandwith, Γ/γ", 22, Black]},
  LabelStyle → Directive[FontFamily → "Times", FontSize → 22, Black],
  PlotLegends → Placed[BarLegend[{ColorData[{"TemperatureMap", "Reverse"}],
      {0, 1}}, LegendLabel → Style["ℬq^cs", 22, Black]], Right],
  FrameTicksStyle → Directive[FontSize → 18, Black]]

savePlot["qBhatCoherent.pdf", Fig1];

(*Export[overleafPathchap4<>"Fig1.png",Fig1];
Export[overleafPathchap4<>"Fig1.pdf",Fig1];
Export[overleafPathchap4<>"Fig1.svg",Fig1];*)
```

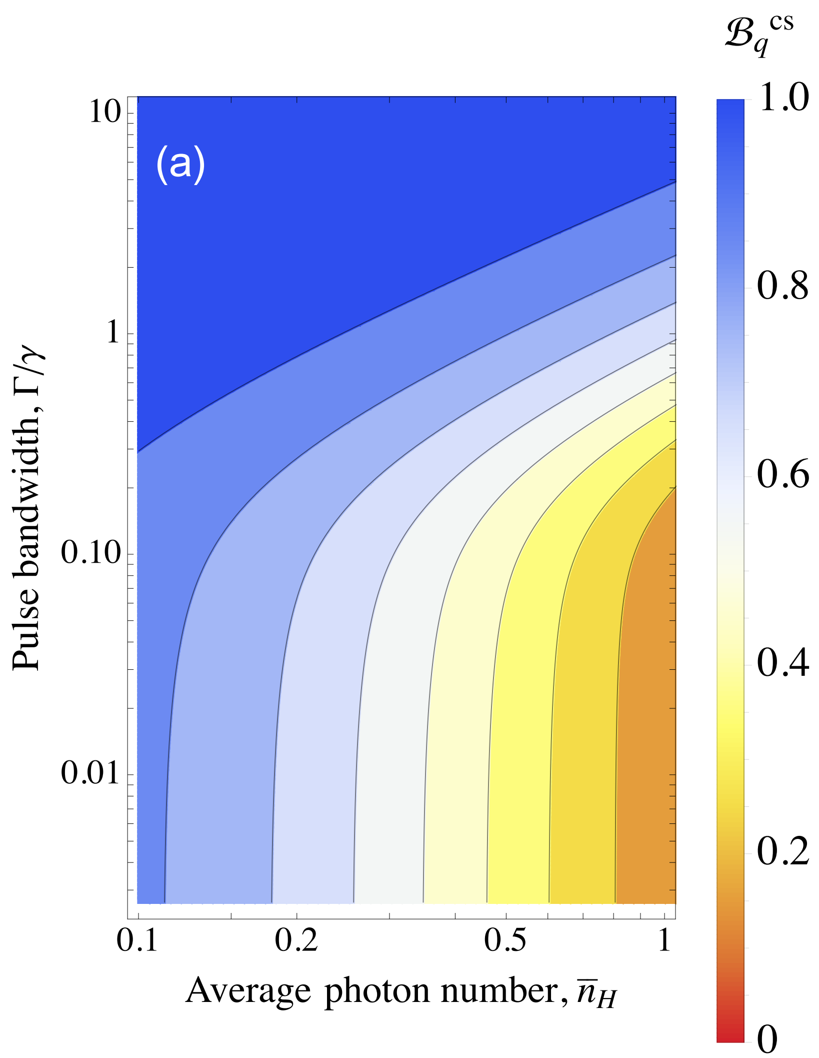
Out[76]=



Cutting out the low energy sector:

In[81]:=
```
lowenergysector = {{All, 1}, {All, 10}};

plotlow = ListContourPlot[coherentsetsemiclassical,
  PlotRange → lowenergysector,
  ScalingFunctions → {"Log", "Log"},
  ColorFunction → colorf,
  ColorFunctionScaling → True, (*Disable color function scaling*)
  FrameLabel → {Style["Average photon number, n̄H", 20, Black],
    Style["Pulse bandwidth, Γ/γ", 20, Black]},
  LabelStyle → Directive[FontFamily → "Times", FontSize → 22, Black],
  PlotLegends → Placed[BarLegend[{ColorData[{"TemperatureMap", "Reverse"}],
      {0, 1}}, LegendLabel → Style["ℬq^cs", 22, Black]], Right],
  Epilog → Inset[Style["(a)", 22, White], Scaled[{0.05, 0.95}], {Left, Top}],
  FrameTicksStyle → Directive[FontSize → 18, Black],
  AspectRatio → 3 / 2,
  ImageSize → 350]
savePlot["LowEnergyqBhatCoherent.pdf", %];
```

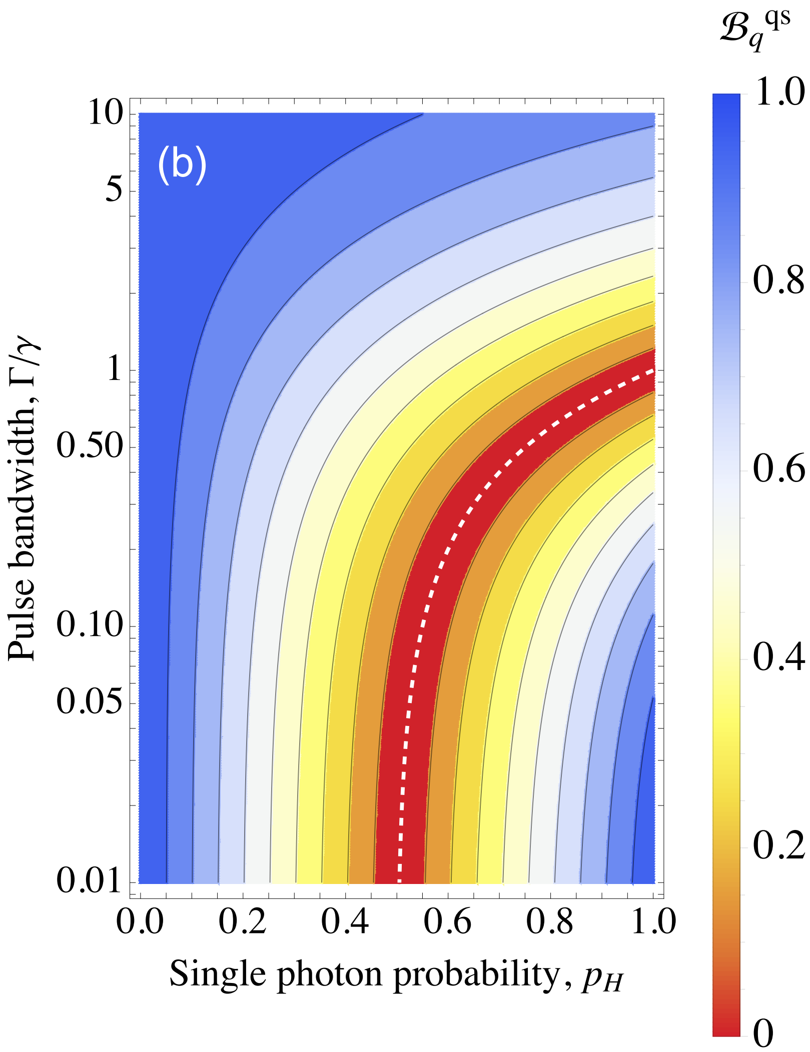
Out[82]=



Defining and plotting the qBhat for the quantum state, ℬqqs:
This is the analytical expression obtained from the model.

In[190]:=
```
ℬqqs = Abs[1 - (2 n γ)/(γ + Γ)];

quantumqBhat = ContourPlot[
   ℬqqs /. γ → 1 /. n → nn /. Γ → ΓΓ, {nn, 0., 1.}, {ΓΓ, 10^-2, 10},
   ColorFunction → ColorData[{"TemperatureMap", "Reverse"}],
   ColorFunctionScaling → False,
   PlotPoints → 300,
   ScalingFunctions → {None, "Log"},
   FrameTicksStyle → Directive["Times", FontSize → 20, Black],
   FrameLabel → {Style["Single photon probability, pH", 20, Black],
     Style["Pulse bandwidth, Γ/γ", 20, Black]},
   LabelStyle → Directive[FontFamily → "Times", FontSize → 22, Black],
   PlotLegends → Placed[BarLegend[{ColorData[{"TemperatureMap", "Reverse"}],
        {0, 1}}, LegendLabel → Style["ℬq^qs", 22, Black]], Right],
   Epilog → Inset[Style["(b)", 22, White], Scaled[{0.05, 0.95}], {Left, Top}],
   AspectRatio → 3 / 2,
   ImageSize → 350];
(*The line where the plot is zero.*)
zerolineOfquantumqBhat = ContourPlot[
   ((1 - (2 n γ)/(γ + Γ)) /. γ → 1 /. n → nn /. Γ → ΓΓ) == 0, {nn, 0., 1.}, {ΓΓ, 10^-2, 10},
   ContourStyle → {Dashed, AbsoluteThickness[2], White},
   PlotPoints → 300,
   ScalingFunctions → {None, "Log"}];
(*Show the contour plot and the zero line*)
qBhatqsPlot = Show[quantumqBhat, zerolineOfquantumqBhat]
(*savePlot["qBhatLowEnergyComparisonPresentation.jpg",%];*)
```

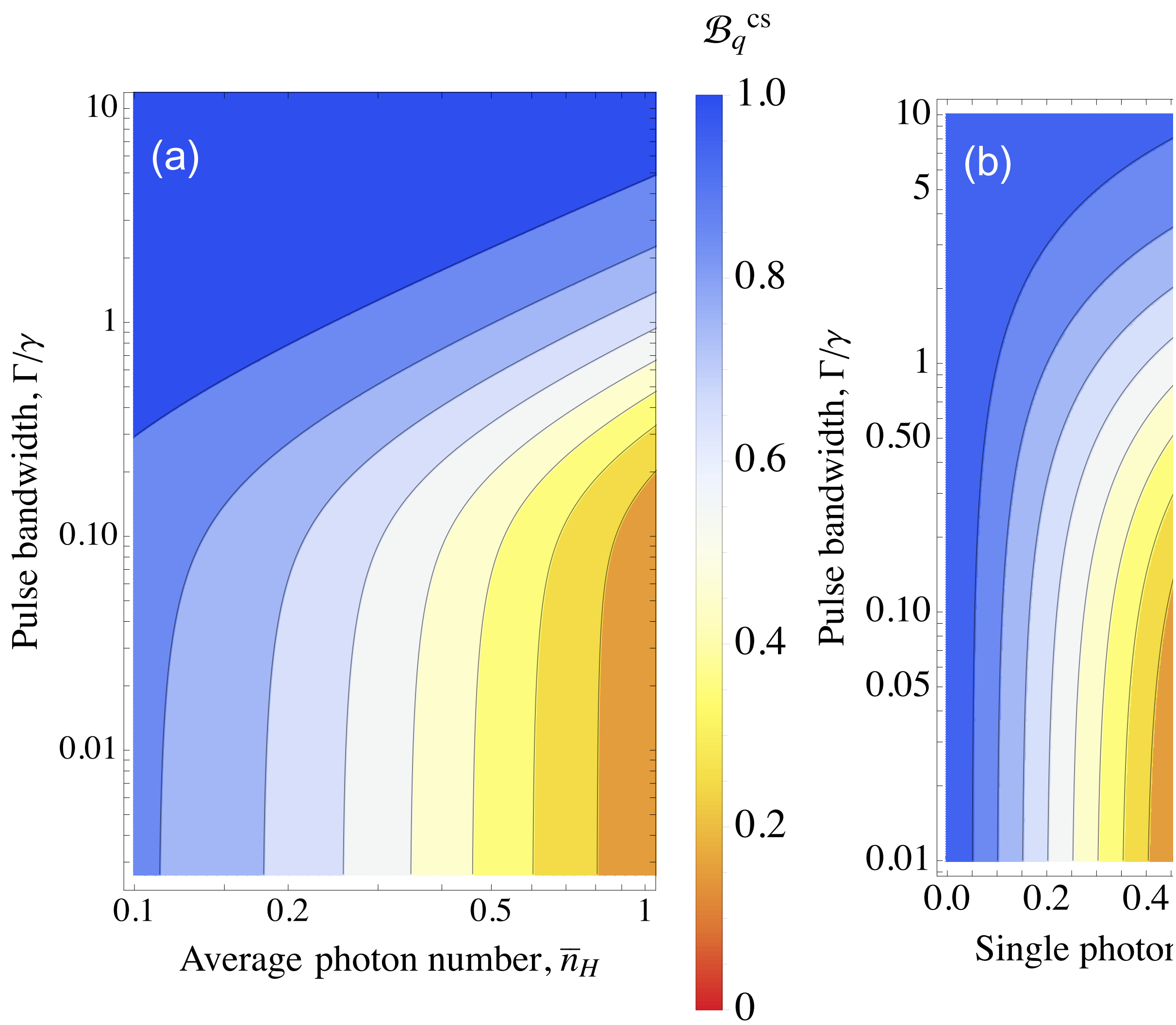
Out[193]=



Show the comparison of the qBhat for the coherent and quantum state:

In[194]:=
```
Fig2 = Grid[{{plotlow, qBhatqsPlot}}]
savePlot["qBhatLowEnergyComparison.pdf", %]
savePlot["qBhatLowEnergyComparison.jpg", %%]
```

Out[194]=



Out[195]= /Users/brunogoes/Dropbox/GitHub/ThesisSPI/Chapter4/PlotsChap4/qBhatLowEnergyComparison.pdf

Out[196]= /Users/brunogoes/Dropbox/GitHub/ThesisSPI/Chapter4/PlotsChap4/qBhatLowEnergyComparison.jpg

Spin read-out:cBhat

Protocol definition

In[197]:=
```
<< cascadedLiouvillian.mx
```

The experiment must be modelled step-by-step. This part of the code was organized by me and is based on the original code written by Dr. Wein. It models the experiment and uses the method

presented in Ref. https://arxiv.org/pdf/2004.04786.pdf .
1) The pulse with one photon passes through the beam splitter:

In[115]:=
```
uL = Cos[θ1] aL;
uR = Cos[θ1] aR;
dL = i Sin[θ1] aL;
dR = i Sin[θ1] aR;
```

2) Part of the pulse will interact with the SPI, hence the Hamiltonian of the light-matter interaction is:

In[119]:=
```
h0 = ΔL σL†.σL + ΔR σR†.σR;
hSPI = -i (Sqrt[ηlm] Sqrt[γ η κ])/2 ((σL†.uL - uL†.σL) + (σR†.uR - uR†.σR));
```

Cascaded coupling Hamiltonian assumes the form:

In[121]:=
```
hcas = h0 + hSPI;
```

3) We write the output modes “u” and “d”:

In[122]:=
```
Loutu = Sqrt[ηd] (Sqrt[γ] σL + Sqrt[η κ] uL);
Routu = Sqrt[ηd] (Sqrt[γ] σR + Sqrt[η κ] uR);

Loutd = Sqrt[ηd] Sqrt[η κ] dL;
Routd = Sqrt[ηd] Sqrt[η κ] dR;
```

4) Passing through the BS again:

In[126]:=
```
RoutD1 = Cos[θ2] Routu + i Exp[i ϕ] Sin[θ2] Routd ;
LoutD1 = Cos[θ2] Loutu + i Exp[i ϕ] Sin[θ2] Loutd ;

RoutD2 = Exp[i ϕ] Cos[θ2] Routd + i Sin[θ2] Routu;
LoutD2 = Exp[i ϕ] Cos[θ2] Loutd + i Sin[θ2] Loutu;
```

5) Defining the (vectorized) Liouvillian:

In[130]:=
```
(*Coherent part*)
PrettyTiming[
 ℋ = unitary2vec[hcas];
 (*Virtual cavity and 4LS decay*)
 𝒟γ = lindblad2vec[Loutu] + lindblad2vec[Routu] + lindblad2vec[Loutd] +
   lindblad2vec[Routd] + (1 - η) κ (lindblad2vec[aR] + lindblad2vec[aL]);
 (*Decay with correlation due to cascaded coupling*)
 𝒟γs = γs lindblad2vec[σL†.σL] + γs lindblad2vec[σR†.σR] +
   κs lindblad2vec[aL†.aL] + κs lindblad2vec[aR†.aR];

 (*Conditional evolution*)
 𝒟ηd =
  -ηRD1 jump[RoutD1] - ηRD2 jump[RoutD2] - ηLD1 jump[LoutD1] - ηLD2 jump[LoutD2];
]
```

0h : 2m : 56s

Total system Liouvillian:

In[131]:=
```
PrettyTiming[ℒcas = ℋ + 𝒟γ + 𝒟γs + 𝒟ηd;]
```

0h : 0m : 7s

In[132]:=
```
(*Checking dimensions and type of the Liouvillian*)
Dimensions@ℒcas
Head@ℒcas
```

Out[132]= {4096, 4096}

Out[133]= List

This takes some time to be built, so let’s save this guy:

In[134]:=
```
DumpSave["cascadedLiouvillian.mx", ℒcas]
```

Out[134]=
```
{{{0. + 0. ⅈ, 0. + 0. ⅈ, 0. + 0. ⅈ, 0. + 0. ⅈ, 0. + 0. ⅈ, 0. + 0. ⅈ, … 4085 …, 0. + 0. ⅈ,
   0. + 0. ⅈ, 0. + 0. ⅈ, 0. + 0. ⅈ, 0. + 0. ⅈ}, … 4094 …, {… 1 …}}}
```

large output | show less | show more | show all | set size limit...

Since the dimension is huge we want to check what are the states that will be explored during the dynamics. To do so, we make an exploration of the fock-Liouville space in what follows.

We put some dummy parameters to get information and use the information that we know the initial states of the field, R, and the 4LS.

Fock-space exploration to dimension reduction

Choose a parameter set to explore the Fock-Liouville space:

In[135]:=
```
par = {
   ΔL → 0, ΔR → 0, (* Detunings *)
   γ → 1, κ → 0.01, (* Decay rates *)
   η → 0.9, ηlm → 0.9, ηd → 0.9, (* Input/coupling efficiency *)
   ηRD1 → 0.5, ηRD2 → 0.5, ηLD1 → 0.5, ηLD2 → 0.5,
   (* Detector efficiency/conditional evolution parameters *)
   γs → 0.1, κs → 0.1, (* Dephasing *)
   ϕ → 0.235, (* interferometer phase *)
   θ1 → 0.412, θ2 → 0.12(* Beam splitter ratios *)
  };
```

Choose an initial state to reduce the necessary subspace: gs spin superposition + H-polarized input coherent state (up to 3 photons):

Is it a number?Did I forget to substitute some variable?

In[136]:=
```
Chop@Total[Flatten[ℒcas /. par]]
```

Out[136]= -3023.86

Choose an initial state to reduce the necessary subspace: gs spin superposition + H-polarized input coherent state (up to 3 photons):

```
In[137]:= istate = Flatten[kp[{1, 1, 0, 0}, {1, 1, 1, 1}, {1, 0, 0, 0}]];
istate = Flatten[kp[istate, istate]];

In[139]:= aux1 = Sign[istate] * Range[dim^2];
(*Sign of initial state assigns 1 to positive values,
0 when the value is 0 and -1 when the value is negative. Range[Dim2] makes a
  big list going from 1 to 4096. The output is a list with a bunch of zeros
  and the indexes of the initial nodes where the values are not zero.*)

In[140]:= initialnodes = DeleteCases[aux1, 0];
(*every place where the value of aux 1 is zero are neglected*)

In[141]:= initialnodes

Out[141]= {1, 5, 9, 13, 17, 21, 25, 29, 257, 261, 265, 269, 273, 277, 281, 285,
 513, 517, 521, 525, 529, 533, 537, 541, 769, 773, 777, 781, 785, 789,
 793, 797, 1025, 1029, 1033, 1037, 1041, 1045, 1049, 1053, 1281, 1285,
 1289, 1293, 1297, 1301, 1305, 1309, 1537, 1541, 1545, 1549, 1553,
 1557, 1561, 1565, 1793, 1797, 1801, 1805, 1809, 1813, 1817, 1821}

In[142]:= aux2 = NumericQ /@ Flatten[ℒcas + x * eye[Length[ℒcas]]] /. {True → 0, False → 1}

Out[142]= {1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, … 16 777 136 …, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1}
  large output | show less | show more | show all | set size limit...

In[143]:= Dimensions@aux2
Head@aux2

Out[143]= {16 777 216}

Out[144]= List

In[145]:= adjacencymatrix = Partition[aux2, dim^2];

In[146]:= Dimensions@adjacencymatrix
Head@adjacencymatrix

Out[146]= {4096, 4096}

Out[147]= List

In[148]:= vertstyle =
  Table[i → If[MemberQ[initialnodes, Range[dim^2][[i]]], Red, Black], {i, dim^2}];
```

In[149]:=
```
aux3 = Transpose[Sign[adjacencymatrix + Transpose[adjacencymatrix]]]
```

Out[149]=
```
{ … 1 … }
large output    show less    show more    show all    set size limit...
```

In[150]:=
```
Dimensions@aux3
Head@aux3
```

Out[150]=
```
{4096, 4096}
```

Out[151]=
```
List
```

In[152]:=
```
AdjacencyGraph[aux3,
 VertexStyle → vertstyle,
 PlotLabel → "Full Fock-Liouville space and intereactions for the ME",
 Frame → True,
 FrameLabel → {"Red dot is the initial state"}]
```

Out[152]=
```
Graph[ Vertex count: 4096
       Edge count: 15360 ]
```

In[153]:=
```
(* This applies the propagator once to see which Fock-
 Liouville states become populated *)
expState = Sign[Abs[MatrixExp[ℒcas /. par].istate]];
```

In[154]:=
```
Dimensions@expState
Head@expState
```

Out[154]=
```
{4096}
```

Out[155]=
```
List
```

In[156]:=
```
aux4 = expState * Range[dim^2]; (*Encontrando os indexes*)
```

In[157]:=
```
(* Let's now look at all the Fock-
 Liouville basis vectors that were explored *)
nodeset = DeleteCases[aux4, 0];
```

In[158]:=
```
nodeset
```

Out[158]=
```
{1, 5, 9, 13, 17, 21, 25, 29, 49, 53, 57, 257, 261, 265, 269, 273, 277, 281, 285,
 305, 309, 313, 513, 517, 521, 525, 529, 533, 537, 541, 561, 565, 569, 769,
 773, 777, 781, 785, 789, 793, 797, 817, 821, 825, 1025, 1029, 1033, 1037,
 1041, 1045, 1049, 1053, 1073, 1077, 1081, 1281, 1285, 1289, 1293, 1297, 1301,
 1305, 1309, 1329, 1333, 1337, 1537, 1541, 1545, 1549, 1553, 1557, 1561,
 1565, 1585, 1589, 1593, 1793, 1797, 1801, 1805, 1809, 1813, 1817, 1821,
 1841, 1845, 1849, 3073, 3077, 3081, 3085, 3089, 3093, 3097, 3101, 3121,
 3125, 3129, 3329, 3333, 3337, 3341, 3345, 3349, 3353, 3357, 3377, 3381,
 3385, 3585, 3589, 3593, 3597, 3601, 3605, 3609, 3613, 3633, 3637, 3641}
```

In[159]:=
```
aux5 = Transpose[Sign[adjacencymatrix + Transpose[adjacencymatrix]]]
```

Out[159]=
```
{ … 1 … }
large output    show less    show more    show all    set size limit...
```

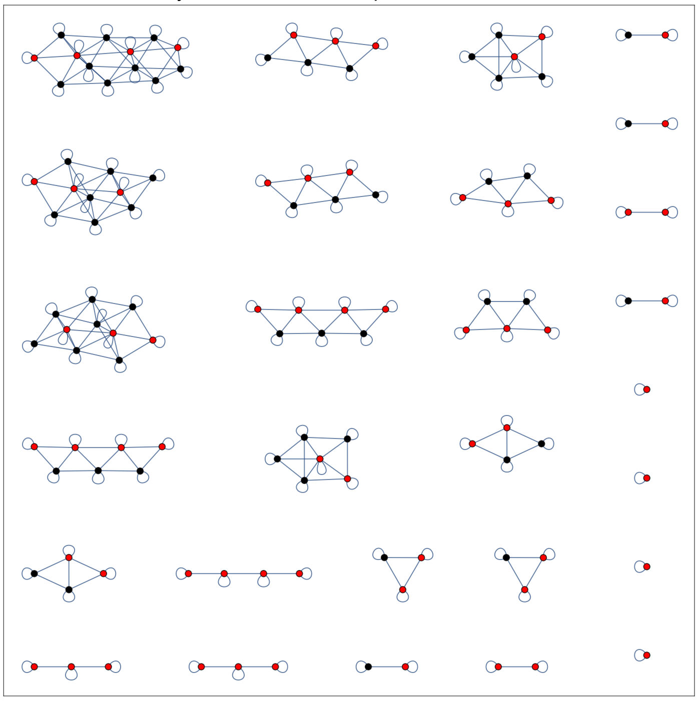
In[160]:=
```
aux6 = aux5[[nodeset, nodeset]]
```

Out[160]=
```
{ … 1 … }
large output    show less    show more    show all    set size limit...
```

In[161]:=
```
AdjacencyGraph[aux6, VertexStyle → Table[
    i → If[MemberQ[initialnodes, nodeset[[i]]], Red, Black], {i, Length[nodeset]}],
 PlotLabel → "Necessary Fock-Liouville subspace for the ME evolution",
 Frame → True, FrameLabel → {"Red dots are the initial state"}]
```

Out[161]=

Necessary Fock–Liouville subspace for the ME evolution



Red dots are the initial state

cBhat simulation

Now that we know which states are accessed durign the dynamics we can restrict the analysis to this smaller subspace:

In[162]:=
```
ℒSS = ℒcas[[nodeset, nodeset]];
initialstateSS = istate[[nodeset]];
traceSS = Flatten[eye[dim]][[nodeset]];
```

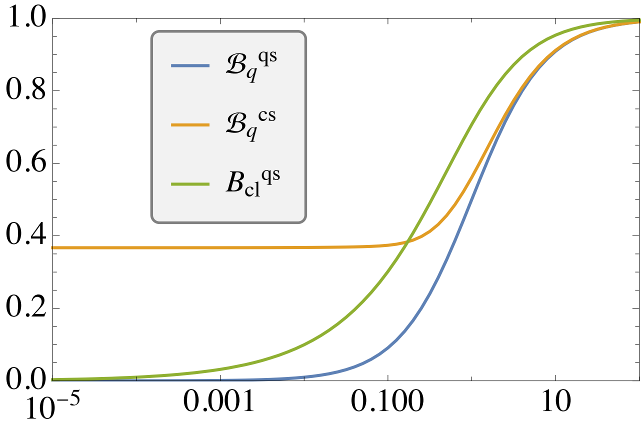
```
In[165]:= Head@ℒSS
          Dimensions@ℒSS
          Head@initialstateSS
          Dimensions@initialstateSS
          Head@traceSS
          Dimensions@traceSS

Out[165]= List

Out[166]= {121, 121}

Out[167]= List

Out[168]= {121}

Out[169]= List

Out[170]= {121}

In[171]:= (* Choose a parameter set to explore the Fock-Liouville space *)
          outset = ParallelTable[
            effN = 1;
            dephN = 0;

            par = {
              ΔL → 0, ΔR → 0, (* Detunings *)
              γ → 1, κ → 10^k, (* Decay rates *)
              η → effN, ηlm → 1, ηd → 1, (* Input/coupling efficiency *)
              ηRD1 → 0., ηRD2 → 0., ηLD1 → 0., ηLD2 → 0.,
              (* Detector efficiency/conditional evolution parameters *)
              γs → dephN, κs → 0 dephN, (* Dephasing *)
              ϕ → 0, (* interferometer phase *)
              θ1 → 0, θ2 → 0
             };

            traceSS = Flatten[eye[dim]]〚nodeset〛;
            σupdw = Outer[Times, spinDw, spinUp];
            σupdw = kp[σupdw, eye[dimSource], eye[dimSource]];
            σupdw = kp[σupdw, eye[dim]]〚nodeset, nodeset〛;

            istate = Flatten[kp[{1, 1, 0, 0}/√2, {1, 1, 0, 0}/√2, {1, 0, 0, 0}]];
            istate = Flatten[kp[istate, istate]]〚nodeset〛;
            phtOut12 = traceSS.σupdw.MatrixPower[MatrixExp[ℒSS /. par], 10^8].istate;

            istate = Flatten[kp[{1, 1, 0, 0}/√2, {0, 1, 0, 0}, {1, 0, 0, 0}]];
            istate = Flatten[kp[istate, istate]]〚nodeset〛;
            phtOut = traceSS.σupdw.MatrixPower[MatrixExp[ℒSS /. par], 10^8].istate;

            istate = Flatten[
```

```
    kp[{1, 1, 0, 0} / Sqrt[2], coherentState[1 / Sqrt[2], 3], {1, 0, 0, 0}]];
  istate = Flatten[kp[istate, istate]]⟦nodeset⟧;
  cohOut12 = traceSS.σupdw.MatrixPower[MatrixExp[ℒSS /. par], 10^8].istate;

  istate = Flatten[kp[{1, 1, 0, 0} / Sqrt[2], coherentState[1, 3], {1, 0, 0, 0}]];
  istate = Flatten[kp[istate, istate]]⟦nodeset⟧;
  cohOut = traceSS.σupdw.MatrixPower[MatrixExp[ℒSS /. par], 10^8].istate;

  par = {
    ΔL → 0, ΔR → 0, (* Detunings *)
    γ → 1, κ → 10^k, (* Decay rates *)
    η → effN, ηlm → 1, ηd → 1, (* Input/coupling efficiency *)
    ηRD1 → 0., ηRD2 → 0., ηLD1 → 0., ηLD2 → 0.,
    (* Detector efficiency/conditional evolution parameters *)
    γs → dephN, κs → 0 dephN, (* Dephasing *)
    ϕ → 0, (* interferometer phase *)
    θ1 → π / 4, θ2 → π / 4
   };

  tsplit = 2;

  Dset = {ηRD1 → 1, ηRD2 → 0, ηLD1 → 1, ηLD2 → 0};

  istate = Flatten[kp[{Cos[θ], Sin[θ], 0, 0}, {0, 1, 0, 0}, {1, 0, 0, 0}]] /. par;
  istate = Flatten[kp[istate, istate]]⟦nodeset⟧;
  prPht = 1 - traceSS.MatrixPower[MatrixExp[ℒSS /. Dset /. par], 10^8].istate;

  {10^k, phtOut12, cohOut12, Sqrt[(prPht /. θ → 0) (prPht /. θ → π / 2)] +
    Sqrt[(1 - prPht /. θ → 0) × (1 - prPht /. θ → π / 2)],
   phtOut, cohOut}, {k, -5, 2, 0.1}];
```

First look at the plot:

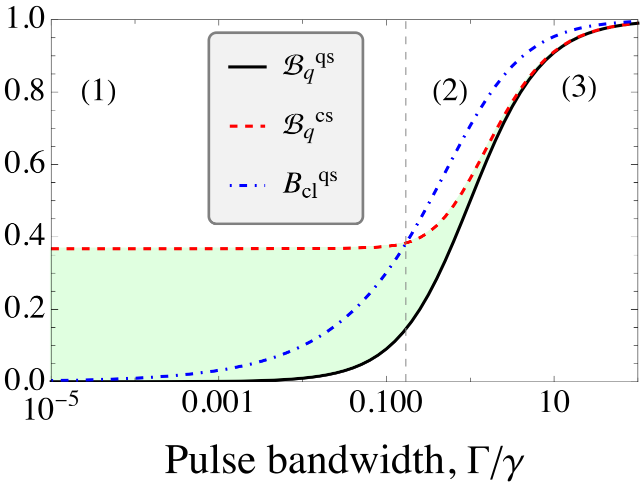
In[172]:=
```
ListPlot[{{#1, 2 Abs[#2]} &@@@ outset, {#1, 2 Abs[#3]} &@@@ outset,
   {#1, #4 // Re} &@@@ outset}, ScalingFunctions → {"Log", None},
  PlotRange → {0., 1},
  Joined → True,
  PlotLegends → legend[{"𝓑q^qs", "𝓑q^cs", "Bcl^qs"}, {0.3, 0.7}]
]
```

Out[172]=



In[179]:=
```
Fig3 = ListPlot[{{#1, 2 Abs[#2]} &@@@ outset, {#1, 2 Abs[#3]} &@@@ outset,
    {#1, #4 // Re} &@@@ outset}, ScalingFunctions → {"Log", None},
  PlotRange → {0., 1},
  Joined → True,
  PlotStyle →
   {Directive[Black], Directive[Dashed, Red], Directive[DotDashed, Blue]},
  PlotLegends → legend[{"𝓑q^qs", "𝓑q^cs", "Bcl^qs"}, {0.4, 0.7}],
  Epilog → {{Dashed, Gray, Line[{{Log10[0.017], 0}, {Log10[0.017], 1}}]},
    Text["(1)", Scaled[{0.05, 0.8}], {-1, 0}],
    Text["(2)", Scaled[{0.68, 0.8}]], , Text["(3)", Scaled[{0.9, 0.81}]]},
  FrameLabel → {Style["Pulse bandwidth, Γ/γ", 22, Black], None},
  Filling → {1 → {2}},
  FillingStyle → LightGreen,
  GridLines → {{Log10[0.017]}, None},
  GridLinesStyle → Directive[Dashed, LightRed]
 ]
savePlot["cBhat.pdf", %];
```

Out[179]=



Out[189]=
```
{1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, … 16 777 136 …, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1}
```

large output | show less | show more | show all | set size limit...