QuESTlink whitepaper

```
Import["https://qtechtheory.org/QuESTlink.m"];
CreateDownloadedQuESTEnv["MacOS"];
```

Demonstrations

Decoherence

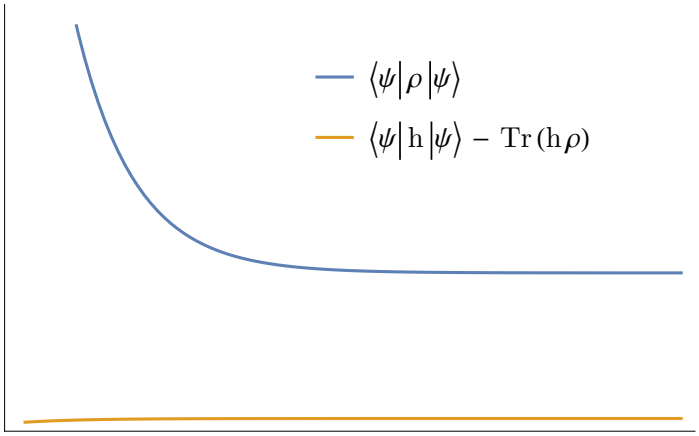
In[4]:=
```
opts = {
    LabelStyle→{FontFamily→"CMU Serif", FontSize→15},
    PlotLegends→Placed[{"⟨ψ| ρ |ψ⟩","⟨ψ| h |ψ⟩ - Tr (h ρ)"}, {{.65,.75}}],
    Frame→{True,True,False,False}, FrameStyle→Black, Axes→None,
    FrameTicks→None, FrameLabel→{"depolarising"}
};
```

In[5]:=
```
{ψ, ϕ} = CreateQuregs[5, 2];
{ρ, σ} = CreateDensityQuregs[5, 2];
SetQuregMatrix[ψ, Normalize @ Table[RandomComplex[], 2^5]];
InitPureState[ρ, ψ];
```

In[9]:=
```
h = .3 + .1 X0 Y1 Z2 - .2 Z0;
data = Table[
    MixTwoQubitDepolarising[ρ, 0, 1, .1];
    {CalcFidelity[ρ, ψ],
     CalcExpecPauliSum[ψ, h, ϕ] -
      CalcExpecPauliSum[ρ, h, σ]},
        100];
```

In[11]:=
```
ListLinePlot[Transpose[data], opts]
```

Out[11]=



depolarising

In[12]:=
```
DestroyAllQuregs[];
```

Variational imaginary time

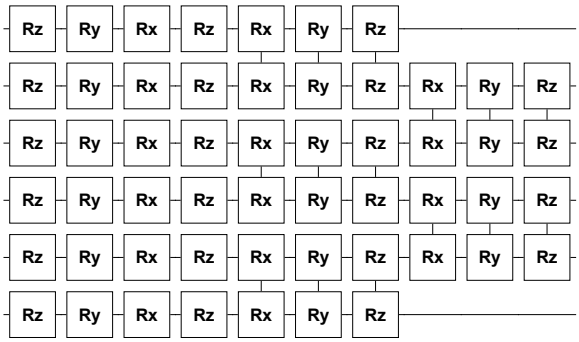
In[13]:=
```
nQb = 6;
h = GetPauliSumFromCoeffs["https://qtechtheory.org/hamil_6qbLiH.txt"];
h[[-2 ;;]]
```

Out[15]= $0.0591748\, Z_1\, Z_2\, Z_4\, Z_5 + 0.147366\, Z_3\, Z_4\, Z_5$

In[16]:=
```
entangle[qbs_] :=
    Table[R[θ, σ_q σ_{q+1}], {σ, {X, Y, Z}}, {q, qbs}]
gates = Flatten @ Join[
    Table[op_{q-1}[θ], {op, {Rz, Ry, Rx, Rz}}, {q, nQb}],
    entangle[{0, 2, 4}],
    entangle[{1, 3}]];
ansatz = MapIndexed[#1 /. θ → θ_{#2[[1]]} &, gates];
nθ = Length[ansatz];
```

In[20]:=
```
DrawCircuit[ansatz, nQb]
```

Out[20]=



In[21]:=
```
{ψ, hψ, ϕ} = CreateQuregs[nQb, 3];
dψ = CreateQuregs[nQb, nθ];
```

In[23]:=
```
curθ = Table[θ_t → RandomReal[], {t, nθ}];
ApplyCircuit[ansatz /. curθ, ψ];
CalcExpecPauliSum[ψ, h, ϕ]
```

Out[25]= -6.98516

In[26]:=
```
Δt = .01;
nt = 100;
Do[
      InitZeroState[ψ];

      CalcQuregDerivs[ansatz, ψ, curθ, dψ];
      matrA = CalcInnerProducts[dψ] // Re;

      ApplyCircuit[ansatz /. curθ, ψ];
      ApplyPauliSum[ψ, h, hψ];
      vecC = -CalcInnerProducts[hψ, dψ] // Re;

      Δθ = Δt LinearSolve[matrA, vecC];
      curθ[[All, 2]] += Δθ,
      nt
 ];
```

In[29]:=
```
CalcExpecPauliSum[ψ, h, ϕ]
```
Out[29]= -7.17522

In[30]:=
```
Min @ Eigenvalues @ CalcPauliSumMatrix @ h
```
Out[30]= -7.88074

In[31]:=
```
energy[θvals__?NumericQ] := Module[{curθ},
  curθ = Table[θ_t → {θvals}[[t]], {t, nθ}];
  InitZeroState[ψ];
  ApplyCircuit[ansatz /. curθ, ψ];
  CalcExpecPauliSum[ψ, h, ϕ]]
```

In[32]:=
```
θvars = Table[θ_t, {t, nθ}];
NMinimize[energy @@ θvars, θvars][[1]]
```
Out[33]= -7.87954

In[34]:=
```
DestroyAllQuregs[];
```

Noisy Trotterisation

In[35]:=
```
opts = {
    Joined→True,PlotStyle→Dashed,PlotMarkers→Graphics`PlotMarkers[][[1]],
    PlotLegends→Placed[
        LineLegend[{"1","2","4","6","8"},LegendMarkerSize→30,Spacings → {1,-.5}],
        {{.85,.4}}],
    PlotRange→{{100,10^5},{0,1}},
    LabelStyle→{FontFamily→"CMU Serif",FontSize→15},
    Frame→True,FrameStyle→{{Black,White},{Black,White}},
    FrameTicks→{{{0,.5,1},None},{Table[{10^n,10^ToString@n},{n,2,5}],None}},
    FrameLabel→{"#gates","Fidelity"},
    Epilog → Text[Style["order:", FontFamily→"CMU Serif",FontSize→15], Scaled[{.85,.8
};
```

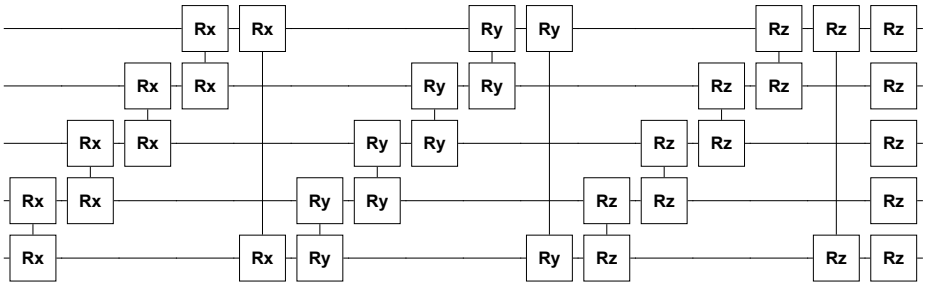
In[36]:=
```
nQb = 5;
h = Flatten @ Join[
    Table[1. σ_{q-1} σ_{Mod[q,nQb]} , {σ, {X, Y, Z}}, {q, nQb}],
    Table[RandomReal[{-1, 1}] Z_{q-1}, {q, nQb}]];
```

In[38]:=
```
symmetrize[h_, λ_, 1] := λ h
symmetrize[h_, λ_, 2] := With[
      {s1 = symmetrize[h, λ/2, 1]},
      Join[s1, Reverse[s1]]]
symmetrize[h_, λ_, n_?EvenQ] := Block[
      {γ, p = 1/(4 - 4^{1/(n-1)})}, With[
      {s = symmetrize[h, γ, n - 2]}, With[
      {r = s /. γ → λ p},
      Join[r, r, s /. γ → (1 - 4 p) λ, r, r]]]]
gateify[Verbatim[Times][θ_, σ__]] :=
     R[2 θ, Times[σ]]
trotterize[h_, n_, r_, t_] := With[
      {s = symmetrize[h, t / r, n]},
       gateify /@ Flatten @ ConstantArray[s, r]]
```

In[43]:=
```
childsify[h_, n_, r_, t_] := Flatten @ Table[
   symmetrize[RandomSample @ h, t / r, n], r]

campbellize[h_, r_, t_] := With[
  {c = h[[All, 1]], σ = h[[All, 2 ;;]], N = Length[h] * r},
    With[{λ = Total[c], p = Abs @ Normalize[c, Total]},
       t * λ/N RandomChoice[p → σ, N]]]
```

In[45]:=
```
DrawCircuit @ trotterize[h, 1, 1, nQb]
```

Out[45]=



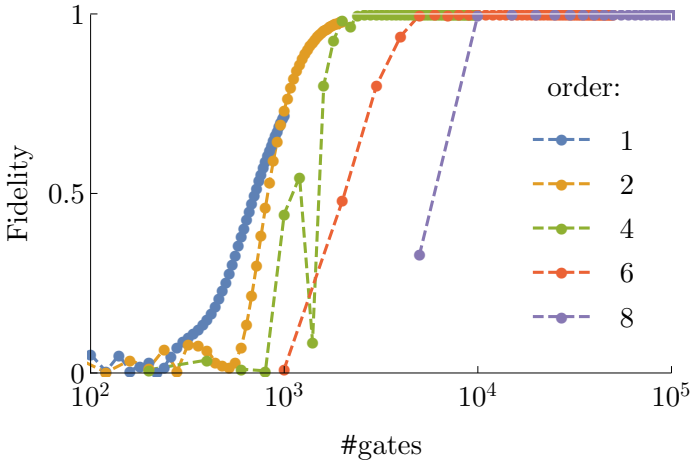
In[46]:=
```
matrify[σ_] :=
    PauliMatrix[σ /. {X → 1, Y → 2, Z → 3}]
matrify[Verbatim[Times][θ_, σ__]] :=
     θ KroneckerProduct @@ Fold[
        ReplacePart[#1, (nQb - #2[[2]]) →
           matrify @ #2[[1]]] &,
        Table[IdentityMatrix[2], nQb], {σ}]

schrod[h_, Ψ0_, t_] := With[
    {H = matrify /@ h // Total},
    NDSolveValue[
        {i Ψ'[τ] == H . Ψ[τ], Ψ[0] == Ψ0}, Ψ,
        {τ, 0, nQb}]][t]
```

In[49]:=
```
ψ0v = Normalize @ Table[RandomComplex[], 2^nQb];
ψ0 = CreateQureg[nQb];
SetQuregMatrix[ψ0, ψ0v];

ψt = CreateQureg[nQb];
SetQuregMatrix[ψt, schrod[h, ψ0v, nQb]];

ψ = CreateQureg[nQb];
CloneQureg[ψ, ψ0];
```

In[56]:=
```
fids = Table[
        circ = trotterize[h, order, reps, nQb];
        ApplyCircuit[circ, ψ0, ψ];
        {Length[circ], CalcFidelity[ψ, ψt]},
        {order, {1, 2, 4, 6, 8}},
        {reps, 1, 50}
  ];
```

In[57]:=
```
ListLogLinearPlot[fids, opts]
```

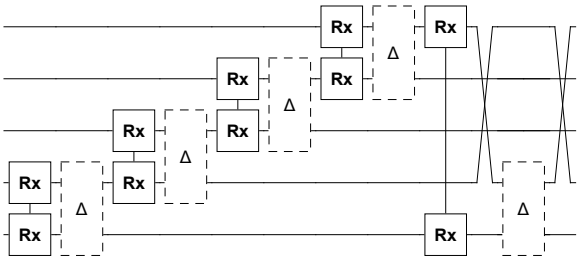
Out[57]=



In[58]:=
```
noisify[p_][u_] := u /. {
    g : R[_, _qb_] :> Sequence[g, Depol_qb[p]],
    g : R[_, Verbatim[Times][_qb1_, _qb2_]] :>
        Sequence[g, Depol_qb1,qb2[p]]}
```

In[59]:=
```
DrawCircuit @ noisify[10^-4] @
    trotterize[h, 1, 1, nQb][[ ;; 5]]
```

Out[59]=

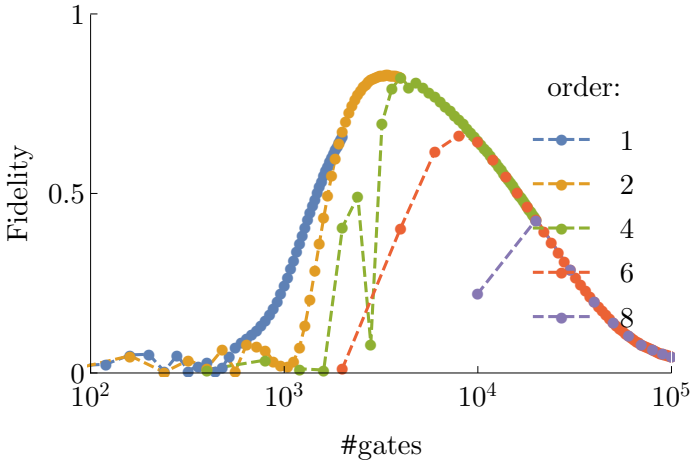


In[60]:=
```
{ρ, ρ0} = CreateDensityQuregs[nQb, 2];
InitPureState[ρ0, ψ0];
```

In[62]:=
```
nfids = Table[
        circ = noisify[10^-4] @
              trotterize[h, order, reps, nQb];
        ApplyCircuit[circ, ρ0, ρ];
        {Length[circ], CalcFidelity[ρ, ψt]},
        {order, {1, 2, 4, 6, 8}},
        {reps, 1, 50}
    ];
```

In[63]:=
```
ListLogLinearPlot[nfids, opts]
```

Out[63]=



In[64]:=
```
DestroyAllQuregs[];
```