1D examples for denoising using wavelets

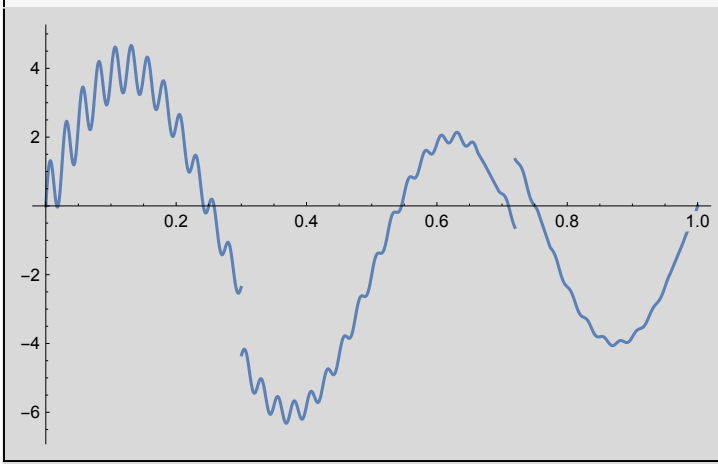
Examples in this notebook use the methods for parameter estimation and transforms provided by the *Mathematica* system

A sample signal

In[259]:=
```
signal[t_] :=
 4 * Sin[4 * Pi * t] + Exp[-3 * t] * Sin[80 * Pi * t] - Sign[t - 0.3] - Sign[0.72 - t]
```

In[260]:=
```
Plot[signal[t], {t, 0, 1}]
```

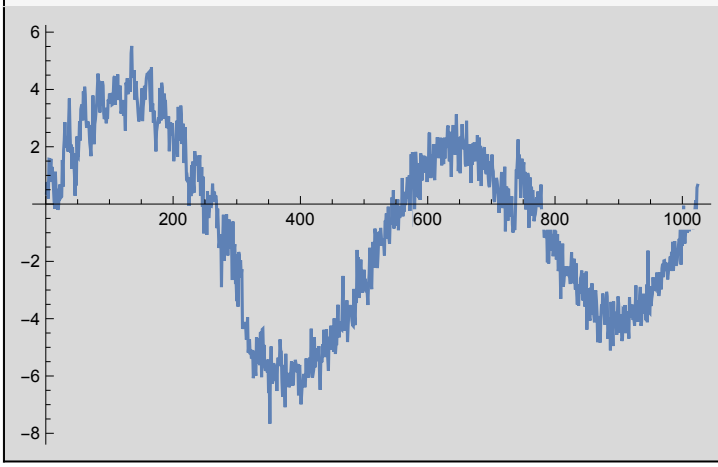
Out[260]=



A noised version of the signal (white noise with σ=0.25)

In[261]:=
```
v = Table[signal[t / 1024], {t, 0, 1023}];
n = Length[v];
nd = NormalDistribution[0., 1.];
SeedRandom[];
noise = Table[Random[nd], {k, 1, n}];
σ = .5;
w = v + σ * noise;
ListPlot[w, Joined → True]
```

Out[268]=



In[269]:=
```
Norm[v - w, 2]
```

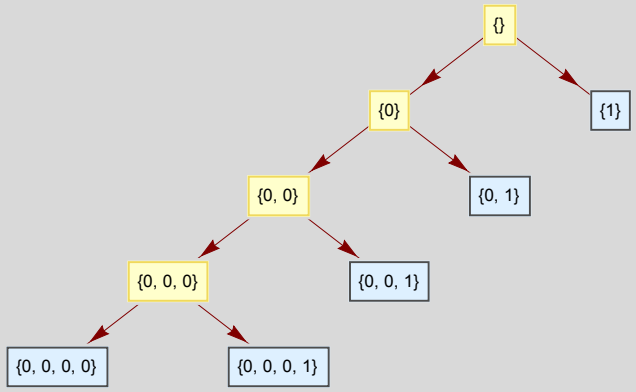
Out[269]=
```
15.4914
```

Denoising using Daubechies-4

In[270]:=
```
dwddaub = DiscreteWaveletTransform[w, DaubechiesWavelet[], 4];
```

In[271]:=
```
dwddaub["TreeView"]
```

Out[271]=



Computing the MAD on the first detail component

In[272]:=
```
σMAD = MedianDeviation[{1} /.dwddaub[{1}]] / 0.6745
```

Out[272]=
```
0.473432
```

Universal parameter based on MAD

In[273]:=
```
λuniv = σMAD √(2 Log[Length[w]])
```

Out[273]=
```
1.76273
```

Sure threshold based on MAD

In[274]:=
```
surethreshold[w_List, σ_] :=
 Block[{n = Length[w], sw, s, id},
  sw = Sort[w^2];
  s = Accumulate[sw] + Range[n - 1, 0, -1] (2 σ^2 + sw);
  id = First[Ordering[s]];
  √sw[[id]] ]
```

In[275]:=
```
λsure = surethreshold[First[dwddaub[{1}, "Values"]], σMAD]
```

Out[275]=
```
1.0351
```

Various types of applying thresholding

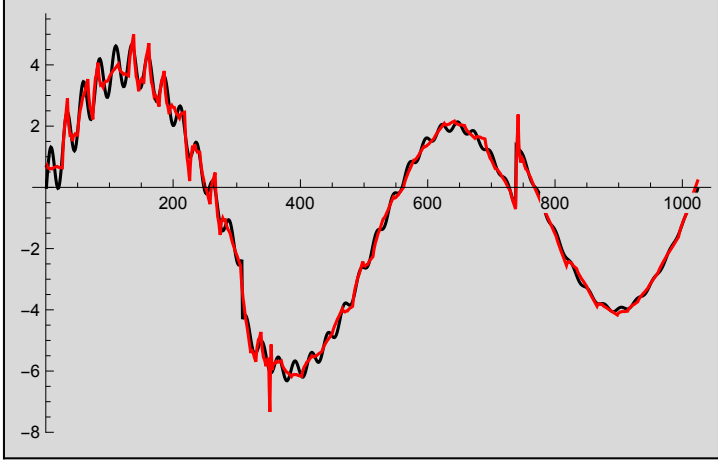
In[276]:=
```
wd1 = WaveletThreshold[dwddaub, {"Hard", λuniv}];
wd2 = WaveletThreshold[dwddaub, {"Hard", λsure}];
wd3 = WaveletThreshold[dwddaub, {"Soft", λuniv}];
wd4 = WaveletThreshold[dwddaub, {"Soft", λsure}];
wd5 = WaveletThreshold[dwddaub, "Universal"];
wd6 = WaveletThreshold[dwddaub, "SURE"];
```

Inverse transforms after thresholding

In[282]:=
```
iwd1 = InverseWaveletTransform[wd1];
iwd2 = InverseWaveletTransform[wd2];
iwd3 = InverseWaveletTransform[wd3];
iwd4 = InverseWaveletTransform[wd4];
iwd5 = InverseWaveletTransform[wd5];
iwd6 = InverseWaveletTransform[wd6];
```

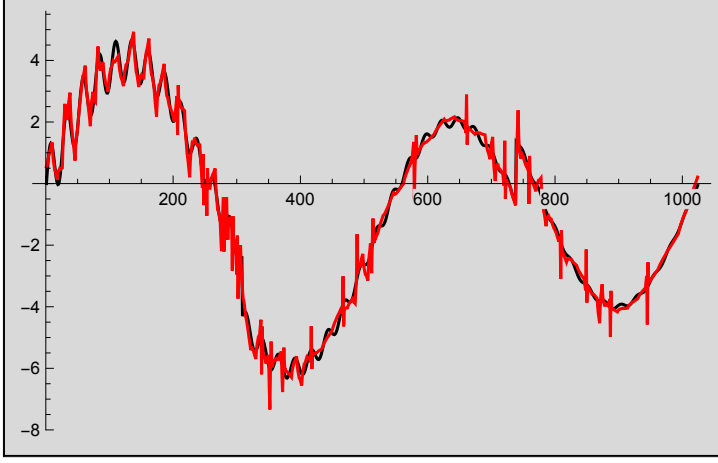
In[288]:=
```
ListLinePlot[{v, iwd1}, PlotRange → All; PlotStyle → {Black, Red}]
```

Out[288]=



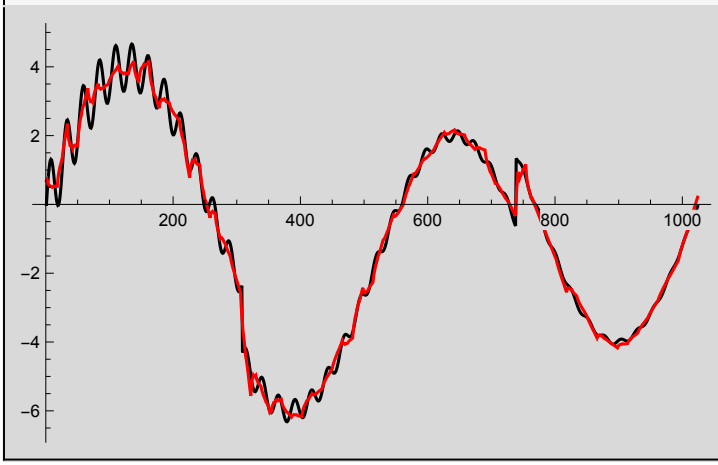
In[289]:=
```
ListLinePlot[{v, iwd2}, PlotRange → All, PlotStyle → {Black, Red}]
```

Out[289]=



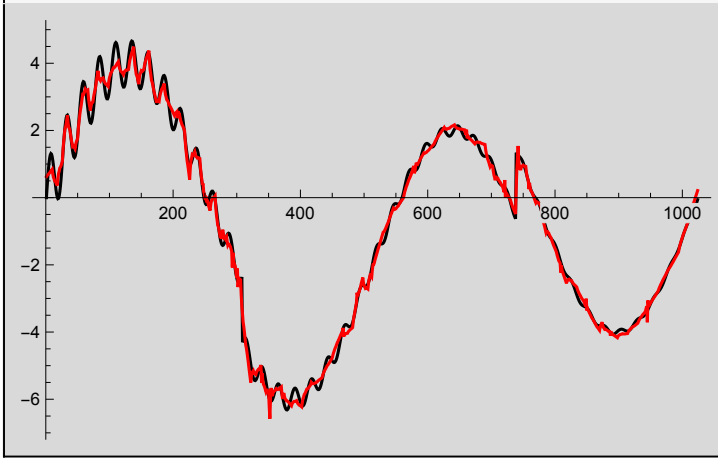
In[290]:=
```
ListLinePlot[{v, iwd3}, PlotRange → All, PlotStyle → {Black, Red}]
```

Out[290]=



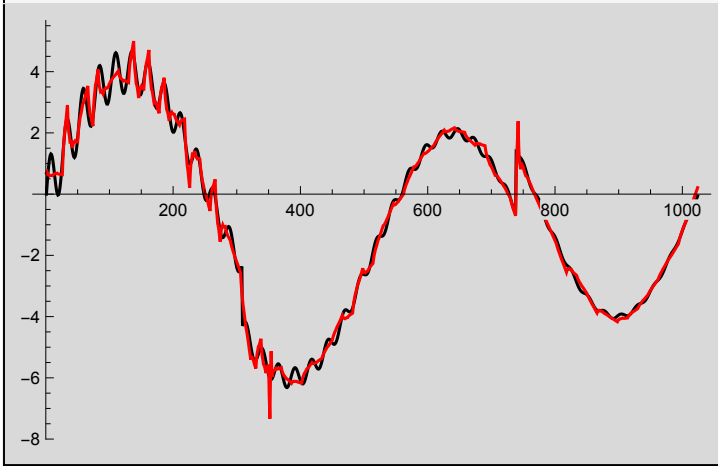
In[331]:=
```
ListLinePlot[{v, iwd4}, PlotRange → All, PlotStyle → {Black, Red}]
```

Out[331]=



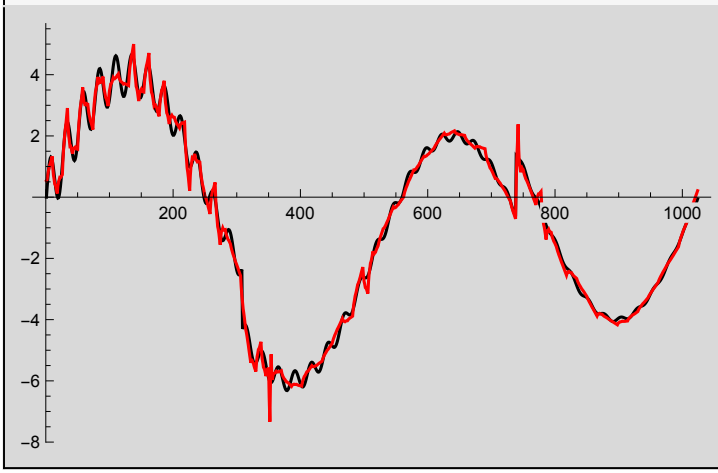
In[291]:=
```
ListLinePlot[{v, iwd5}, PlotRange → All, PlotStyle → {Black, Red}]
```

Out[291]=



In[292]:=
```
ListLinePlot[{v, iwd6}, PlotRange → All, PlotStyle → {Black, Red}]
```

Out[292]=



In[293]:=
```
Map[Norm[# - v, 2] &, {iwd1, iwd2, iwd3, iwd4, iwd5, iwd6}]
```

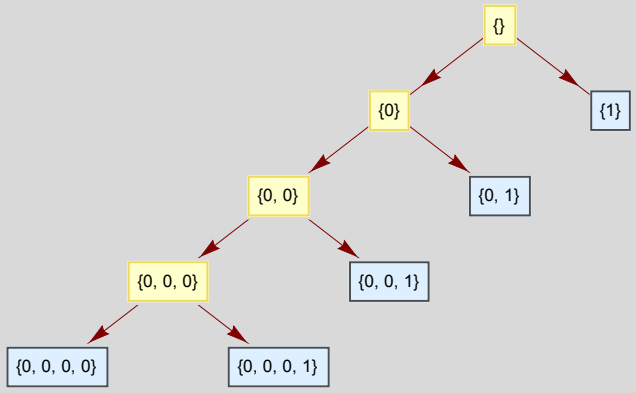
Out[293]=
```
{7.72398, 9.20476, 8.54206, 6.98773, 7.72398, 7.11953}
```

Denoising using Haar

In[294]:=
```
dwdhaar = DiscreteWaveletTransform[w, HaarWavelet[], 4];
```

In[295]:=
```
dwdhaar["TreeView"]
```

Out[295]=



In[296]:=
```
λsure = surethreshold[First[dwdhaar[{1}, "Values"]], σMAD]
```

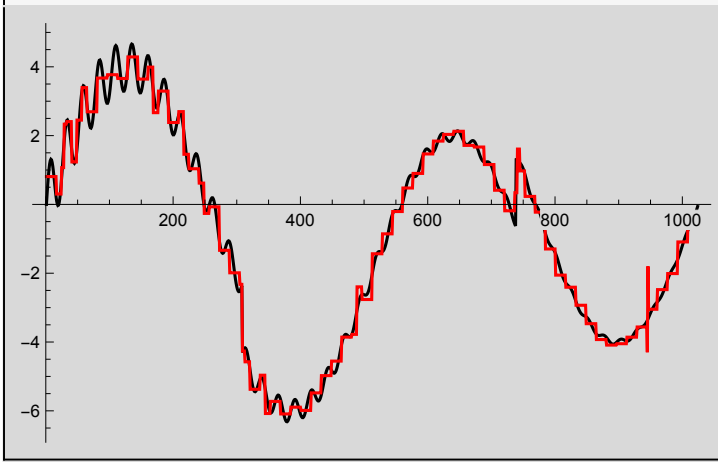
Out[296]=
```
0.956076
```

In[297]:=
```
wd7 = WaveletThreshold[dwdhaar, "Universal"];
```

In[298]:=
```
iwd7 = InverseWaveletTransform[wd7];
```

In[299]:=
```
ListLinePlot[{v, iwd7}, PlotRange → All, PlotStyle → {Black, Red}]
```

Out[299]=



In[300]:=
```
Norm[iwd7 - v, 2]
```

Out[300]=
```
9.05123
```

In[301]:=
```
wd8 = WaveletThreshold[dwdhaar, "SURE"];
```

In[302]:=
```
iwd8 = InverseWaveletTransform[wd8];
```

In[303]:=
```
Norm[iwd8 - v, 2]
```

Out[303]=
```
8.99482
```

Denoising using Coiflet-12

In[304]:=
```
dwdsym2 = DiscreteWaveletTransform[w, CoifletWavelet[2]];
```

In[305]:=
```
λsure = surethreshold[First[dwdsym2[{1}, "Values"]], σMAD]
```

Out[305]=
```
1.01863
```

In[306]:=
```
wd9 = WaveletThreshold[dwdsym2, "Universal"];
```

In[307]:=
```
iwd9 = InverseWaveletTransform[wd9];
```

In[308]:=
```
Norm[iwd9 - v, 2]
```

Out[308]=
```
7.85165
```

In[309]:=
```
dwdsym4 = DiscreteWaveletTransform[w, CoifletWavelet[4]];
```

In[310]:=
```
λsure = surethreshold[First[dwdsym4[{1}, "Values"]], σMAD]
```

Out[310]=
```
1.00124
```

In[311]:=
```
wd10 = WaveletThreshold[dwdsym4, "SURE"];
```

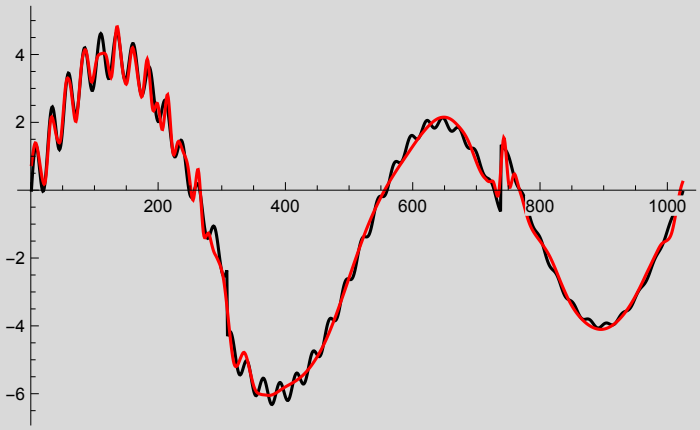
In[312]:=
```
iwd10 = InverseWaveletTransform[wd10];
```

In[313]:=
```
Norm[iwd10 - v, 2]
```

Out[313]=
```
7.82042
```

In[314]:=
```
ListLinePlot[{v, iwd10}, PlotRange → All, PlotStyle → {Black, Red}]
```

Out[314]=

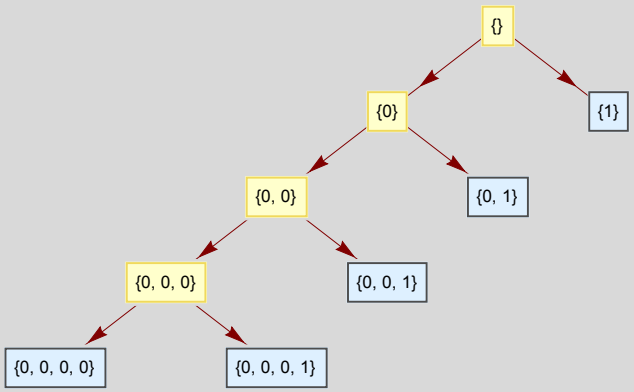


Denoising using a biorthogonal spline wavelet

In[315]:=
```
dwdbio24 = DiscreteWaveletTransform[w, BiorthogonalSplineWavelet[2, 4], 4];
```

In[316]:=
```
dwdbio24["TreeView"]
```

Out[316]=



In[317]:=
```
λsure = surethreshold[First[dwdbio24[{1}, "Values"]], σMAD]
```

Out[317]=
```
0.905093
```

In[318]:=
```
wd11 = WaveletThreshold[dwdbio24, "SURE"];
```

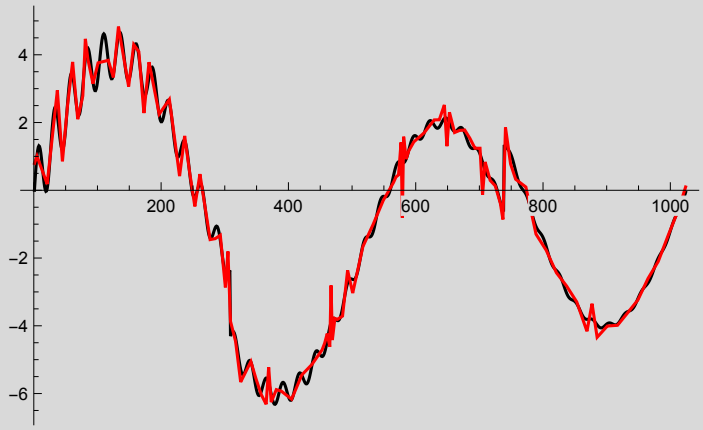
In[319]:=
```
iwd11 = InverseWaveletTransform[wd11];
```

In[320]:=
```
Norm[iwd11 - v, 2]
```

Out[320]=
```
7.38031
```

In[321]:=
```
ListLinePlot[{v, iwd11}, PlotRange → All, PlotStyle → {Black, Red}]
```

Out[321]=



In[322]:=
```
dwdbio35 = DiscreteWaveletTransform[w, BiorthogonalSplineWavelet[3, 5], 3];
```

In[323]:=
```
λsure = surethreshold[First[dwdbio35[{1}, "Values"]], σMAD]
```

Out[323]=
```
1.24991
```

In[324]:=
```
wd12 = WaveletThreshold[dwdbio35, "Universal"];
```

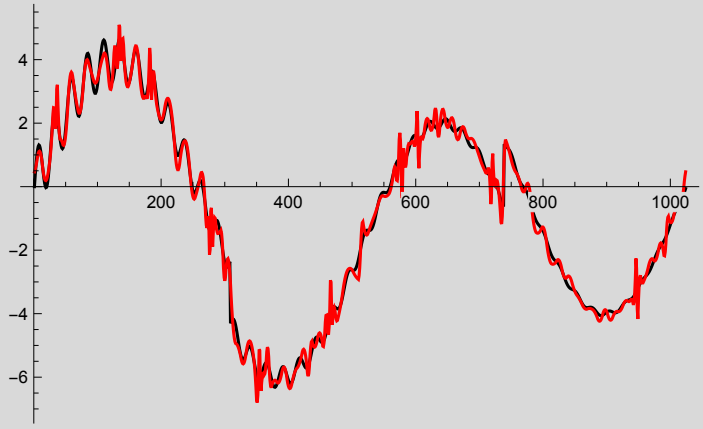
In[325]:=
```
iwd12 = InverseWaveletTransform[wd12];
```

In[326]:=
```
Norm[iwd12 - v, 2]
```

Out[326]=
```
8.44982
```

In[327]:=
```
ListLinePlot[{v, iwd12}, PlotRange → All, PlotStyle → {Black, Red}]
```

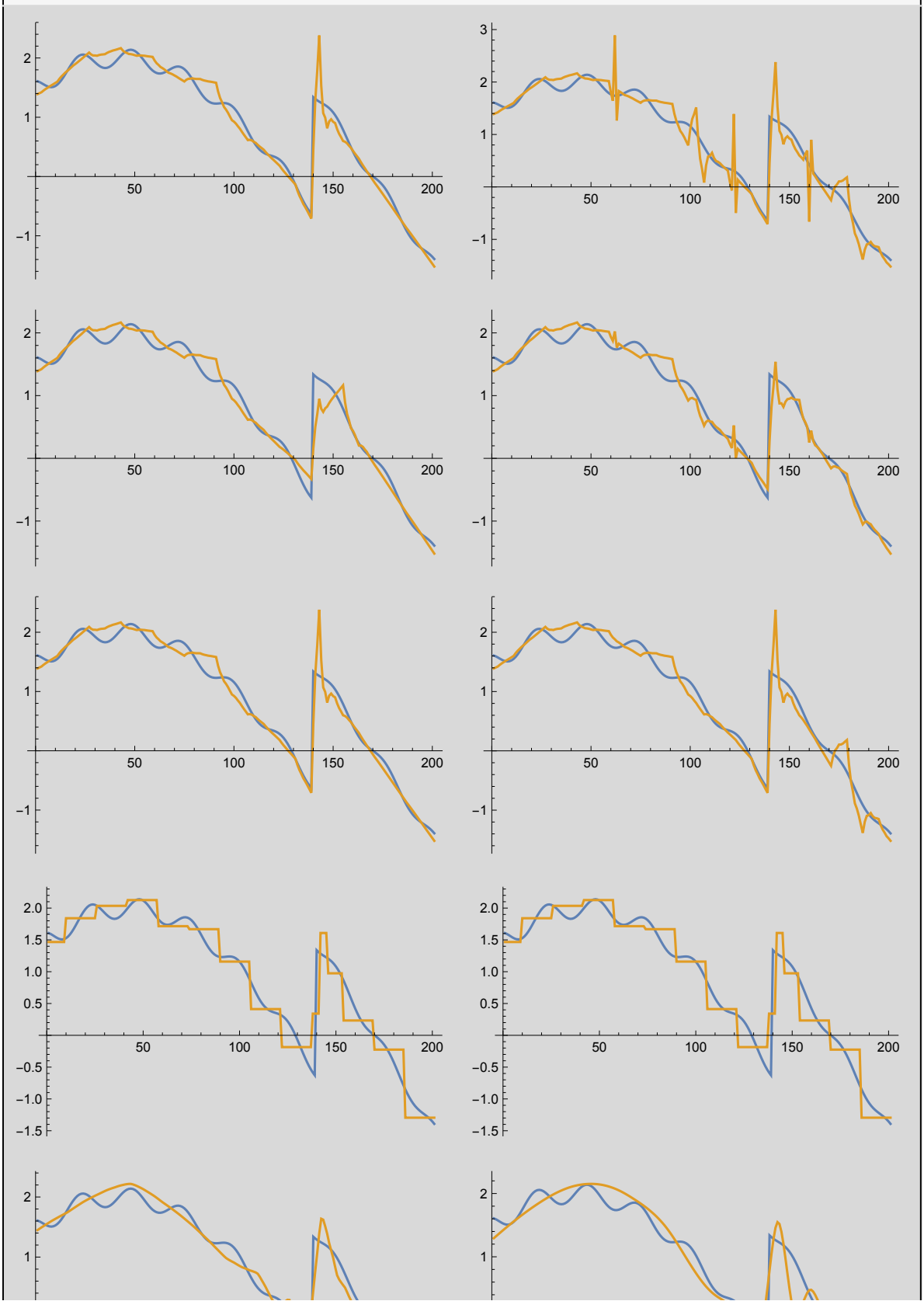
Out[327]=

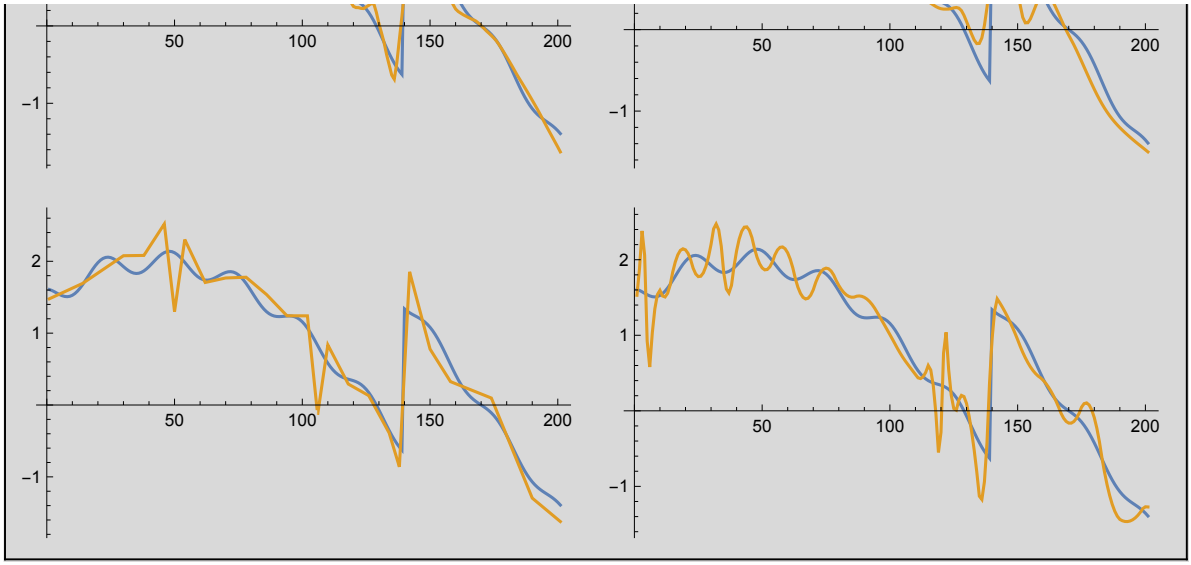


In[328]:=

```
GraphicsGrid[
 Partition[
  Map[
   ListLinePlot[{v[[600 ;; 800]], #[[600 ;; 800]]}] &, {iwd1, iwd2, iwd3, iwd4,
    iwd5, iwd6, iwd7, iwd8, iwd9, iwd10, iwd11, iwd12}], 2], ImageSize → Full]
```

Out[328]=





In[329]:=

```
ass = Association[
  1 → {"D4", "univ-hard"},
  2 → {"D4", "sure-hard"},
  3 → {"D4", "univ-soft"},
  4 → {"D4", "sure-soft"},
  5 → {"D4", "Universal"},
  6 → {"D4", "Sure"},
  7 → {"Haar", "Universal", 4},
  8 → {"Haar", "SURE", 4},
  9 → {"C12", "Universal", 4},
  10 → {"C12", "SURE", 4},
  11 → {"Bio24", "SURE", 3},
  12 → {"Bio35", "SURE", 3}]
```

Out[329]=

```
⟨| 1 → {D4, univ-hard}, 2 → {D4, sure-hard}, 3 → {D4, univ-soft},
 4 → {D4, sure-soft}, 5 → {D4, Universal}, 6 → {D4, Sure},
 7 → {Haar, Universal, 4}, 8 → {Haar, SURE, 4}, 9 → {C12, Universal, 4},
 10 → {C12, SURE, 4}, 11 → {Bio24, SURE, 3}, 12 → {Bio35, SURE, 3} |⟩
```

In[330]:=
```
Table[{"iwd" <> ToString[k], Norm[ToExpression["iwd" <> ToString[k]] - v, 2],
   k /. ass}, {k, 1, 12}] // TableForm
```

Out[330]//TableForm=

iwd1	7.72398	D4 univ-hard
iwd2	9.20476	D4 sure-hard
iwd3	8.54206	D4 univ-soft
iwd4	6.98773	D4 sure-soft
iwd5	7.72398	D4 Universal
iwd6	7.11953	D4 Sure
iwd7	9.05123	Haar Universal 4
iwd8	8.99482	Haar SURE 4
iwd9	7.85165	C12 Universal 4
iwd10	7.82042	C12 SURE 4
iwd11	7.38031	Bio24 SURE 3
iwd12	8.44982	Bio35 SURE 3