Spectrum of trace of N-layer Transfer Matrix

Joaquin Garcia-Suarez (2021)

This notebook contains the calculations, as well as code to generate figures, presented in pre-print titled “Trace spectrum of 1D Transfer Matrices for wave propagation in layered media”.

Preliminaries

Fix some parameters

Frequency range to take into consideration:

```
In[5]:= fmin = 0.1; fmax = 12.5;
```

Set work directory (wherever the Kiknet txt files are):

```
directoryName = "/home/...";
SetDirectory[directoryName];
```

Data access

Data downloaded from https://www.kyoshin.bosai.go.jp/cgi-bin/kyoshin/db/sitedat.cgi?1+N-MRH04+kik ... etc.

The 10 sites to be considered in this study (included in the release):

```
In[8]:= SitesList = {"fksh14.txt", "fksh11.txt", "iwth08.txt", "iwth27.txt", "ksrh06.txt",
    "ksrh07.txt", "nigh11.txt", "nigh14.txt", "nmrh04.txt", "tkch08.txt"};
```

```
In[9]:= (*Export["table_example.pdf",Framed[Style[Import[SitesList[[2]]],FontSize→14]]]*)
```

Create site name list:

```
In[10]:= ListLocNames =
   Table[StringSplit[ToUpperCase[SitesList[[index]]], "."][[1]], {index, 1, 10}];
```

Example of info contained in the txt file:

In[11]:=
```
Framed[Style[Import[SitesList[[-1]]], FontSize → 14]]
```

Out[11]=

No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	4.00,	4.00,	300.00,	130.00
2,	32.00,	36.00,	1850.00,	480.00
3,	42.00,	78.00,	1850.00,	590.00
4,	-----,	-----,	5000.00,	2800.00

Writing the exact expressions:

Trace of a laminate

Choose number of layers in the laminate:

In[12]:=
```
NN = 3;
```

Generation from the cosine products:

In[13]:=
```
(*maximum number of tangents in any expression*)
maxNumberTerms = Floor[NN, 2];
(*number of factors in each group of expressions*)
numberTerms = 2 # & /@ Range[Floor[NN, 2]/2];
(*number of addends belonging to each group*)
numberAddends = Binomial[NN, #] & /@ numberTerms;
indexVectors = If[NN > #[[1]],
      Join[ConstantArray[1, #[[1]]], ConstantArray[0, NN - #[[1]]]],
      ConstantArray[1, #[[1]]]
    ] & /@ Transpose@{numberTerms, numberAddends};
indexSets = Flatten[Permutations[#] & /@ indexVectors, 1];
anTF = 1;
Do[
  layers = Flatten@Position[indexSets[[term]], 1];
  Zs = Sqrt[ρ# * μ#] & /@ layers;
  sortedZs = Zs[[# ;; ;; 2]] & /@ {1, 2};
  anTF = anTF + (-1)^(Length[layers]/2) * 1/2 *
     (Times @@ sortedZs[[1]]/Times @@ sortedZs[[2]] + Times @@ sortedZs[[2]]/Times @@ sortedZs[[1]]) Times @@ (Tan[r#] & /@ layers);
  , {term, 1, Length@indexSets}];
anTF = (Times @@ (Cos[r#] & /@ Range[NN])) * anTF;
Framed[anTF]
```

Out[21]=

$$\boxed{\mathrm{Cos}[r_1]\,\mathrm{Cos}[r_2]\,\mathrm{Cos}[r_3]\left(1-\frac{1}{2}\left(\frac{\sqrt{\mu_1\,\rho_1}}{\sqrt{\mu_2\,\rho_2}}+\frac{\sqrt{\mu_2\,\rho_2}}{\sqrt{\mu_1\,\rho_1}}\right)\mathrm{Tan}[r_1]\,\mathrm{Tan}[r_2]-\frac{1}{2}\left(\frac{\sqrt{\mu_1\,\rho_1}}{\sqrt{\mu_3\,\rho_3}}+\frac{\sqrt{\mu_3\,\rho_3}}{\sqrt{\mu_1\,\rho_1}}\right)\mathrm{Tan}[r_1]\,\mathrm{Tan}[r_3]-\frac{1}{2}\left(\frac{\sqrt{\mu_2\,\rho_2}}{\sqrt{\mu_3\,\rho_3}}+\frac{\sqrt{\mu_3\,\rho_3}}{\sqrt{\mu_2\,\rho_2}}\right)\mathrm{Tan}[r_2]\,\mathrm{Tan}[r_3]\right)}$$

Generation from the cosine sum (compare to eq.(18) for instance):

In[22]:=
```
nAddends = 2^(NN-1); (*different coefficients and different cosine arguments*)
(*Possible combinations for the coefficient -- impedance constrats --*)
Ccombos = {ConstantArray[0, NN]};
Do[
  Ccombos =
   Join[Ccombos, Permutations[Join[ConstantArray[1, 2 * ii], ConstantArray[0, NN - 2 ii]]]]
  , {ii, 1, Floor[NN / 2]}];
(*Combination for the argument of the cosines*)
Scombos = Join[{1}, #] & /@
   Permutations[Join[ConstantArray[1, NN - 1], ConstantArray[-1, NN - 1]], {NN - 1}];
```

```
cyclesMatrix = Table[Scombos[[kk]].Ccombos[[ii]], {ii, 1, nAddends}, {kk, 1, nAddends}];
(*Create the coefficients to generate the impedance contrasts*)
impeExps = {};
Do[
  thisCombo = Ccombos[[ii]];
  If[Total[thisCombo] == 0, impeExps = AppendTo[impeExps, ConstantArray[0, NN]],
   AppendTo[impeExps, {1, 0, 0, 0}];
   auxFlag = 1;
   (*the first term is always 1*)
   Do[
    Which[
     thisCombo[[jj]] == 1 && auxFlag == 1, (impeExps[[-1]][[jj]] = -1;
      auxFlag = 0), (**)
     thisCombo[[jj]] == 1 && auxFlag == 0, (impeExps[[-1]][[jj]] = +1;
      auxFlag = 1),
     thisCombo[[jj]] == 0 && auxFlag == 0, impeExps[[-1]][[jj]] = 0,
     thisCombo[[jj]] == 0 && auxFlag == 1, impeExps[[-1]][[jj]] = 0
    ]
    , {jj, 2, NN}]
  ]
  , {ii, 1, Length@Ccombos}];
(*Create impedance contrasts... all of them!*)
impeLists = {1};
Do[
  layers = Flatten@Position[Ccombos[[term]], 1];
  Zs = Sqrt[ρ# * μ#] & /@ layers;
  sortedZs = Zs[[# ;; ;; 2]] & /@ {1, 2};
  AppendTo[impeLists,
   (-1)^(Length[layers]/2) * 1/2 (Times @@ sortedZs[[1]] / Times @@ sortedZs[[2]] + Times @@ sortedZs[[2]] / Times @@ sortedZs[[1]])];
  , {term, 2, Length@Ccombos}];
discreteSpectrum =
  Table[Total[Scombos[[ii]][[#]] * r# & /@ Range[NN]], {ii, 1, Length@Scombos}];
cosineList = Cos[#] & /@ discreteSpectrum;
amplitudes =
  Table[Total[Table[impeLists[[ii]] * (-1)^(cyclesMatrix[[ii, kk]]/2), {ii, 1, Length@impeLists}]],
   {kk, 1, Length@Scombos}];
anTFv2 = (amplitudes.cosineList) / 2^(NN-1);
Framed[anTFv2]
```

Out[35]=

$$\frac{1}{4}\left(\text{Cos}[r_1+r_2-r_3]\left(1+\frac{1}{2}\left(\frac{\sqrt{\mu_1\,\rho_1}}{\sqrt{\mu_2\,\rho_2}}+\frac{\sqrt{\mu_2\,\rho_2}}{\sqrt{\mu_1\,\rho_1}}\right)+\frac{1}{2}\left(-\frac{\sqrt{\mu_1\,\rho_1}}{\sqrt{\mu_3\,\rho_3}}-\frac{\sqrt{\mu_3\,\rho_3}}{\sqrt{\mu_1\,\rho_1}}\right)+\frac{1}{2}\left(-\frac{\sqrt{\mu_2\,\rho_2}}{\sqrt{\mu_3\,\rho_3}}-\frac{\sqrt{\mu_3\,\rho_3}}{\sqrt{\mu_2\,\rho_2}}\right)\right)+\right.$$
$$\text{Cos}[r_1-r_2+r_3]\left(1+\frac{1}{2}\left(-\frac{\sqrt{\mu_1\,\rho_1}}{\sqrt{\mu_2\,\rho_2}}-\frac{\sqrt{\mu_2\,\rho_2}}{\sqrt{\mu_1\,\rho_1}}\right)+\frac{1}{2}\left(\frac{\sqrt{\mu_1\,\rho_1}}{\sqrt{\mu_3\,\rho_3}}+\frac{\sqrt{\mu_3\,\rho_3}}{\sqrt{\mu_1\,\rho_1}}\right)+\frac{1}{2}\left(-\frac{\sqrt{\mu_2\,\rho_2}}{\sqrt{\mu_3\,\rho_3}}-\frac{\sqrt{\mu_3\,\rho_3}}{\sqrt{\mu_2\,\rho_2}}\right)\right)+$$
$$\text{Cos}[r_1-r_2-r_3]\left(1+\frac{1}{2}\left(-\frac{\sqrt{\mu_1\,\rho_1}}{\sqrt{\mu_2\,\rho_2}}-\frac{\sqrt{\mu_2\,\rho_2}}{\sqrt{\mu_1\,\rho_1}}\right)+\frac{1}{2}\left(-\frac{\sqrt{\mu_1\,\rho_1}}{\sqrt{\mu_3\,\rho_3}}-\frac{\sqrt{\mu_3\,\rho_3}}{\sqrt{\mu_1\,\rho_1}}\right)+\frac{1}{2}\left(\frac{\sqrt{\mu_2\,\rho_2}}{\sqrt{\mu_3\,\rho_3}}+\frac{\sqrt{\mu_3\,\rho_3}}{\sqrt{\mu_2\,\rho_2}}\right)\right)+$$
$$\left.\text{Cos}[r_1+r_2+r_3]\left(1+\frac{1}{2}\left(\frac{\sqrt{\mu_1\,\rho_1}}{\sqrt{\mu_2\,\rho_2}}+\frac{\sqrt{\mu_2\,\rho_2}}{\sqrt{\mu_1\,\rho_1}}\right)+\frac{1}{2}\left(\frac{\sqrt{\mu_1\,\rho_1}}{\sqrt{\mu_3\,\rho_3}}+\frac{\sqrt{\mu_3\,\rho_3}}{\sqrt{\mu_1\,\rho_1}}\right)+\frac{1}{2}\left(\frac{\sqrt{\mu_2\,\rho_2}}{\sqrt{\mu_3\,\rho_3}}+\frac{\sqrt{\mu_3\,\rho_3}}{\sqrt{\mu_2\,\rho_2}}\right)\right)\right)$$

Verification and plots

In[147]:=

```
DE = 0.; (*No need to introduce damping in these calculations*)
(*For the figures:*)
freqTicks = Table[{ii, ToString[ii*10]}, {ii, 1, fmax}];
transferFunctionFigures = Table[{}, {Length[SitesList]}];
profileFigures = Table[{}, {Length[SitesList]}];
dispFigures = Table[{}, {Length[SitesList]}];
spectrumFigures = Table[{}, {Length[SitesList]}];
Quiet@Do[(*Print[jj];*)
  (*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*)
  (*Import and Prepare
    Data -----------------------------------------------------------*)
  data = Import[SitesList[[jj]], "Table"];
  SiteName = StringSplit[ToUpperCase[SitesList[[jj]]], "."][[1]];
  Nrows = Extract[1][Dimensions[data]];
  (*Layer depth, average depth and thickness*)
  Depths =
   Flatten[Table[ToExpression[StringSplit[data〚ii, 3〛, ","]], {ii, 3, Nrows - 1}]];
  AvDepths = Join[{Depths[[1]]/2}, Table[(Depths[[ii]] + Depths[[ii - 1]])/2,
     {ii, 2, Length[Depths]}]];
  hs = Flatten[Table[ToExpression[StringSplit[data〚ii, 2〛, ","]], {ii, 3, Nrows - 1}]];
  Htotal = Total[hs];
  (*The list is ordered from free surface to bottom*)
  (*Layer shear-wave velocities, densities, impedances and fundamental
   period estimate -----------------------------------------------*)
  Vs = Flatten[Table[data〚ii, 5〛, {ii, 3, Nrows}]];
```

```
Vbase = Vs[[-2]];
rhos = Table[1500., {ii, 1, Length[Vs]}];
mus = Table[rhos[[ii]] * Vs[[ii]]^2, {ii, 1, Length[Vs]}];
alphas = Table[rhos[[ii]] * Vs[[ii]] / (rhos[[ii + 1]] * Vs[[ii + 1]]), {ii, 1, Nrows - 3}];
(*Frequency interval*)
flist = Subdivide[fmin, fmax, 200];
(*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*)
(*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*)
(*Compute Transfer Function
 (Aki and Richard's) -------------------------------------*)
layerMatrix = hs[[#]] * {{0., 1 / (mus[[#]] * (1 + I * DE))}, {-ω^2 * rhos[[#]], 0.}} & /@
   Range[Length@hs];
(*Layer matrices*)
exactExpFreq = {};
Do[
 (*Create the exponential matrices
  for each layer and put them in order to multiply them*)
 (*expMatList=Apply[MatrixExp,f[layerMatrix[[#]]/.ω→2*π*flist[[ii]]]]&/@
    Range[Length@hs];*)
 expMatList = func[layerMatrix[[#]] /. ω → 2 * π * flist[[ii]]] & /@
    Range[Length@hs] /. func → MatrixExp;
 (*Proceed with the multiplication*)
 exactExp = expMatList[[1]];
 Do[exactExp = expMatList[[jj]].exactExp,
  {jj, 2, Length@hs}];
 (*Add to the list of values*)
 exactExpFreq =
  AppendTo[exactExpFreq, {flist[[ii]], 0.5 * (exactExp[[1, 1]] + exactExp[[2, 2]])}]
 , {ii, 1, Length@flist}];
(*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*)
(*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*)
(*Compute and Evaluate Analytical ---------
 ----------------------------------------------------*)
NN = Length@hs;
(*The other
  formula -----------------------------------------------------------*)
```

```
nAddends = 2^(NN-1); (*different coefficients and different cosine arguments*)
(*Possible combinations for the coefficient -- impedance constrats --*)
Ccombos = {ConstantArray[0, NN]};
Do[
 Ccombos = Join[Ccombos,
   Permutations[Join[ConstantArray[1, 2*ii], ConstantArray[0, NN - 2 ii]]]]
 , {ii, 1, Floor[NN/2]}];
(*Combination for the argument of the cosines*)
Scombos = Join[{1}, #] & /@
  Permutations[Join[ConstantArray[1, NN - 1], ConstantArray[-1, NN - 1]], {NN - 1}];
cyclesMatrix = Table[Scombos[[kk]].Ccombos[[ii]], {ii, 1, nAddends}, {kk, 1, nAddends}];
(*Create the coefficients to generate the impedance contrasts*)
impeExps = {};
Do[
 thisCombo = Ccombos[[ii]];
 If[Total[thisCombo] == 0, impeExps = AppendTo[impeExps, ConstantArray[0, NN]],
  AppendTo[impeExps, {1, 0, 0, 0}];
  auxFlag = 1;
  (*the first term is always 1*)
  Do[
   Which[
    thisCombo[[jj]] == 1 && auxFlag == 1, (impeExps[[-1]][[jj]] = -1;
     auxFlag = 0), (**)
    thisCombo[[jj]] == 1 && auxFlag == 0, (impeExps[[-1]][[jj]] = +1;
     auxFlag = 1),
    thisCombo[[jj]] == 0 && auxFlag == 0, impeExps[[-1]][[jj]] = 0,
    thisCombo[[jj]] == 0 && auxFlag == 1, impeExps[[-1]][[jj]] = 0
   ]
   , {jj, 2, NN}]
 ]
 , {ii, 1, Length@Ccombos}];
(*Create impedance contrasts ... all of them!*)
impeLists = {1};
Do[
 layers = Flatten@Position[Ccombos[[term]], 1];
 Zs = Sqrt[ρ# * μ#] & /@ layers;
 sortedZs = Zs[[# ;; ;; 2]] & /@ {1, 2};
 AppendTo[impeLists,
  (-1)^(Length[layers]/2) * 1/2 (Times @@ sortedZs[[1]] / Times @@ sortedZs[[2]] + Times @@ sortedZs[[2]] / Times @@ sortedZs[[1]])];
 , {term, 2, Length@Ccombos}];
discreteSpectrum =
```

```
 Table[Total[Scombos[[ii]][[#]] * r_# & /@ Range[NN]], {ii, 1, Length@Scombos}];
cosineList = Cos[#] & /@ discreteSpectrum ;
amplitudes =
 Table[Total[Table[impeLists[[ii]] * (-1)^(cyclesMatrix [[ii,kk]]/2), {ii, 1, Length@impeLists}]],
  {kk, 1, Length@Scombos}];
anTFv2 = (amplitudes .cosineList) / 2^(NN-1);
(*Prepare to evaluate numerically... the amplitudes and the periods*)
amplitudesNum = amplitudes /. μ1 ρ1 → (rhos[[1]] * Vs[[1]])^2;
periodsNum = discreteSpectrum /. r1 → hs[[1]] / Vs[[1]];
Do[
 auxVar2 = amplitudesNum ;
 auxVar3 = periodsNum ;
 amplitudesNum = auxVar2 /. μi ρi → (rhos[[i]] * Vs[[i]])^2;
 periodsNum = auxVar3 /. ri → hs[[i]] / Vs[[i]];
 , {i, 2, NN}];
amplitudesNum = 1 / 2^(NN-1) * amplitudesNum ;
(*Prepare to evaluate numerically*)
auxVar = (anTFv2 /. r1 → 2 π * f * hs[[1]] / (Vs[[1]] * Sqrt[1 + ⅈ * DE])) /. μ1 ρ1 → (rhos[[1]] * Vs[[1]])^2;
Do[
 auxVar2 = auxVar;
 auxVar = (auxVar2 /. ri → 2 π * f * hs[[i]] / (Vs[[i]] * Sqrt[1 + ⅈ * DE])) /. μi ρi → (rhos[[i]] * Vs[[i]])^2;
 , {i, 2, NN}];
auxVar = Expand[auxVar];
(*Make table evaluating numerically*)
flist2 = Subdivide[0, fmax, 10 000];
myTFv2 = Table[{flist2[[k]], auxVar /. f → flist2[[k]]}, {k, 1, Length@flist2}];
(*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*)
(*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*)
(*Plot Half trace ------------------------------------------------------------*)
TFPlot = ListLinePlot[{
   exactExpFreq ,
   myTFv2},
```

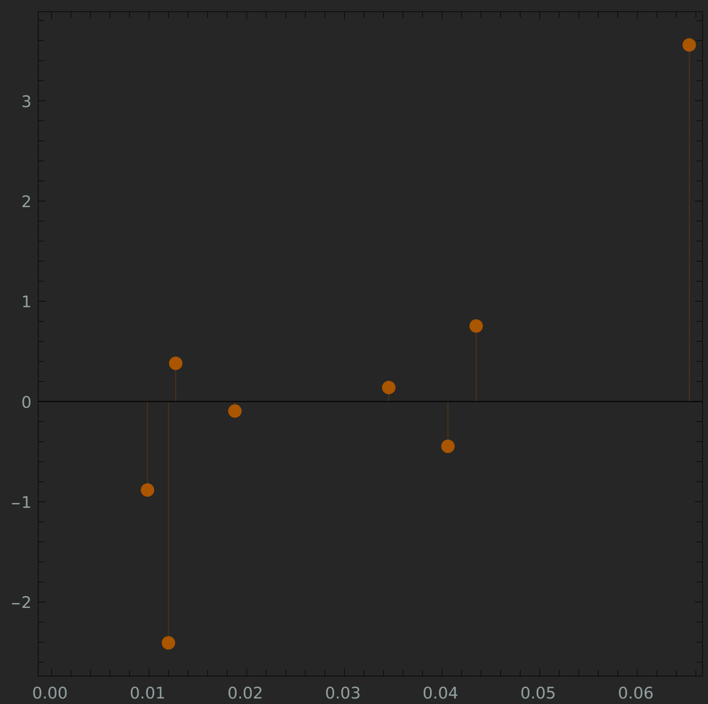
```
  PlotRange → {{fmin, fmax}, {Full, Full}},
  Axes → False,
  GridLines → {None, {{1, Dashed}, {-1, Dashed}}},
  PlotStyle → {{Automatic, Thickness[0.012]},
    {Orange, Dashed, Thickness[0.015/3]}},
  Frame → {{True, False}, {True, False}},
  FrameTicks → {{Automatic, None}, {freqTicks, None}},
  PlotLegends → Placed[
    LineLegend[Style[#, FontSize → 16] & /@ {"Transfer function", "Analytical"},
     LegendLayout → "Column"], {If[jj == 1 || jj == 4, Right, Left], Top}],
  RotateLabel -> False];
transferFunctionFigures[[jj]] = TFPlot;
(*Plot Evolution*)
Vticks = Table[{Vs[[ii]], Rotate[Vs[[ii]], 90 Degree]}, {ii, Length[Vs] - 1}];
Zticks = Table[{-Depths[[ii]], Depths[[ii]]}, {ii, Length[Depths]}];
ProfilePlot = ParametricPlot[{Piecewise[
    Table[{Vs[[ii]], z < Depths[[ii]]}, {ii, 1, Length[Vs] - 1}]], -z}, {z, 0, Depths[[-1]]},
  PlotRange → {{0, Max[Vs[[1 ;; -2]]] * 1.1}, {0, -Depths[[-1]]}},
  Frame → True,
  FrameTicks → {{Zticks, None}, {Vticks, None}},
  AspectRatio → 2];
profileFigures[[jj]] = ProfilePlot;
(*Plot Spectrum ------------------------------------------------------------*)
colorFun = Function[{x, y},
  Blend[{{Min[amplitudesNum], Yellow}, {Max[amplitudesNum], Red}}, Norm[{x, y}]]];
halfSpectrum = Transpose@{Abs[periodsNum], amplitudesNum};
spectraTableFigures =
 ListPlot[halfSpectrum,
  Axes → {True, False},
  Frame → True,
  Filling → Axis,
  PlotStyle → {Darker@Orange, PointSize[Large]},
  ColorFunction → colorFun,
  AspectRatio → 1]
;
spectrumFigures[[jj]] = spectraTableFigures;
(*Plot Dispersion
  Relation -------------------------------------------------------------*)
dispFunction = Interpolation[
  Table[{myTFv2[[kk]][[1]], ArcCos[Re@myTFv2[[kk]][[2]]]}, {kk, 1, Length@myTFv2}]
 ];
```

```
  dispFunction2 = Interpolation[
     Table[{myTFv2[[kk]][[1]], Re@myTFv2[[kk]][[2]]}, {kk, 1, Length@myTFv2}]
   ];
  dispRel =
   Table[{ArcCos[Re@myTFv2[[kk]][[2]]], myTFv2[[kk]][[1]]}, {kk, 1, Length@myTFv2}];
  drTableFigures =
   ParametricPlot[{dispFunction[x], x}, {x, 0. , fmax},
     PlotRange → {{0, π}, {fmin, fmax}},
     PlotStyle → {Darker@Red, Thickness[0.015/1.5]},
     AspectRatio → 1.5,
     Axes → False,
     Frame → True,
     FrameTicks → {{freqTicks, None}, {{{0.05, 0}, {0.99 * π, π}}, None}}];
  dispFigures[[jj]] = drTableFigures;
  ,
  {jj, 1, Length@SitesList}];
```

Examples:

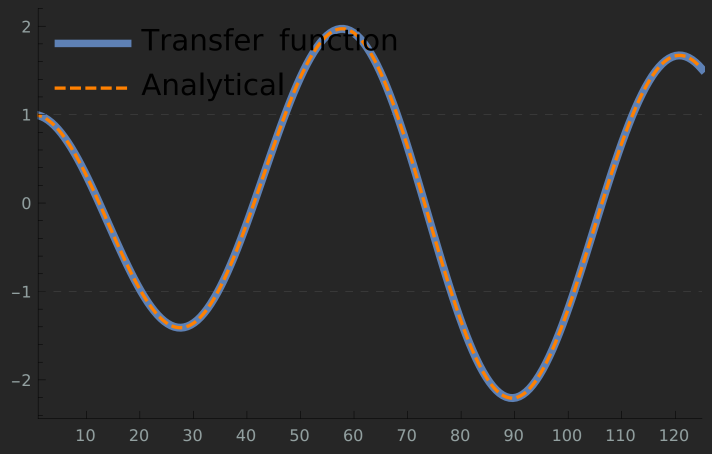
In[154]:= spectrumFigures[[4]]

Out[154]=



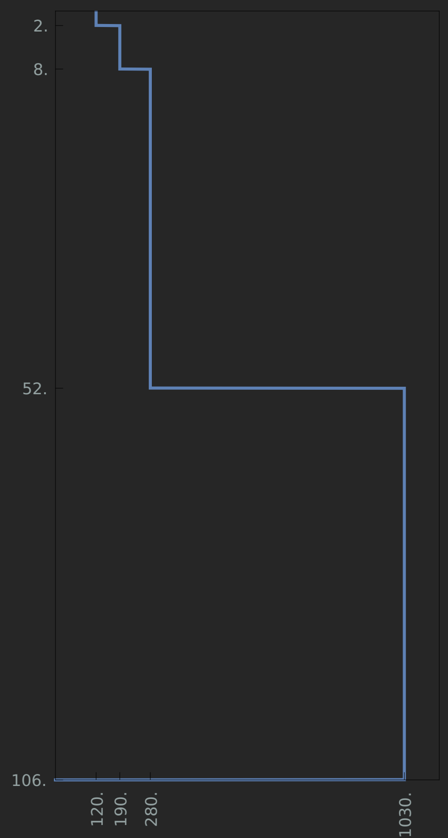
In[155]:= transferFunctionFigures [[-1]]

Out[155]=



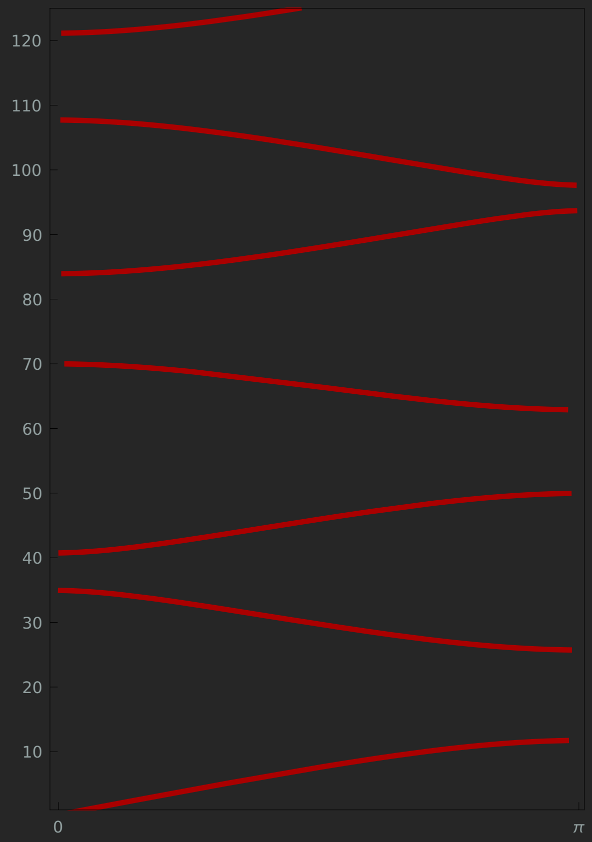
In[156]:= profileFigures [[1]]

Out[156]=



In[157]:=
```
dispFigures[[6]]
```

Out[157]=



Design example

In[158]:=
```
DE = 0;
(*The design
 values ---------------------------------------------------------------------------------*)
r1 = 0.15 * 2;
r2 = 0.15;
r3 = 0.15;
(*The physical
 values----------------------------------------------------------------------------------*)
(*The list is ordered from free surface to bottom*)
(*Layer shear-wave velocities, densities, impedances and fundamental
 period estimate ---------------------------------------------------*)
Vs = {100., 300., 900.};
hs = {r1 * Vs[[1]], r2 * Vs[[2]], r3 * Vs[[3]]};
Depths = Table[Total[hs[[1 ;; ii]]], {ii, 1, Length@hs}];
rhos = Table[1500, {ii, 1, Length[Vs]}];
mus = Table[rhos[[ii]] * Vs[[ii]]^2, {ii, 1, Length[Vs]}];
```

```
alphas = Table[rhos[[ii]] * Vs[[ii]] / (rhos[[ii + 1]] * Vs[[ii + 1]]), {ii, 1, Length[Vs] - 1}];
Z1 = 0.5 * (alphas[[1]] + 1 / alphas[[1]]);
Z2 = 0.5 * (alphas[[2]] + 1 / alphas[[2]]);
Vticks = Table[{Vs[[ii]], Rotate[Vs[[ii]], 90 Degree]}, {ii, Length[Vs]}];
Zticks =
  Table[{-Depths[[ii]], ToString @ NumberForm[Depths[[ii]], {3, 0}]}, {ii, Length[Depths]}];
(*Frequency interval*)
flist = Subdivide[fmin, fmax, 200];
(*Compute Transfer Function
 (Aki and Richard's) ---------------------------------------*)
layerMatrix = hs[[#]] * {{0., 1 / (mus[[#]] * (1 + i * 0.))}, {-ω^2 * rhos[[#]], 0.}} & /@
   Range[Length @ hs];(*Layer matrices*)
exactExpFreq = {};
Do[
  (*Create the exponential matrices
   for each layer and put them in order to multiply them*)
  (*expMatList =Apply[MatrixExp ,f[layerMatrix [[#]]/ .ω→2*π*flist[[ii]]]]&/@
     Range[Length@hs];*)
  expMatList = func[layerMatrix [[#]] /. ω → flist[[ii]]] & /@ Range[Length @ hs] /.
    func → MatrixExp ;
  (*Proceed with the multiplication *)
  exactExp = expMatList [[1]];
  Do[exactExp = expMatList [[jj]].exactExp ,
   {jj, 2, Length @ hs}];
  (*Add to the list of values*)
  exactExpFreq =
   AppendTo[exactExpFreq , {flist[[ii]], 0.5 * (exactExp [[1, 1]] + exactExp [[2, 2]])}]
  , {ii, 1, Length @ flist}];
interpExactExpFreq = Interpolation [exactExpFreq ];
(*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%    *)
(*Compute and Evaluate Analytical ---------
 -----------------------------------------------------*)
NN = Length @ hs;
(*maximum number of tangents in any expression*)
maxNumberTerms = Floor[NN, 2];
(*number of factors in each group of expressions*)
numberTerms = 2 # & /@ Range[Floor[NN, 2] / 2];
(*number of addends belonging to each group*)
```

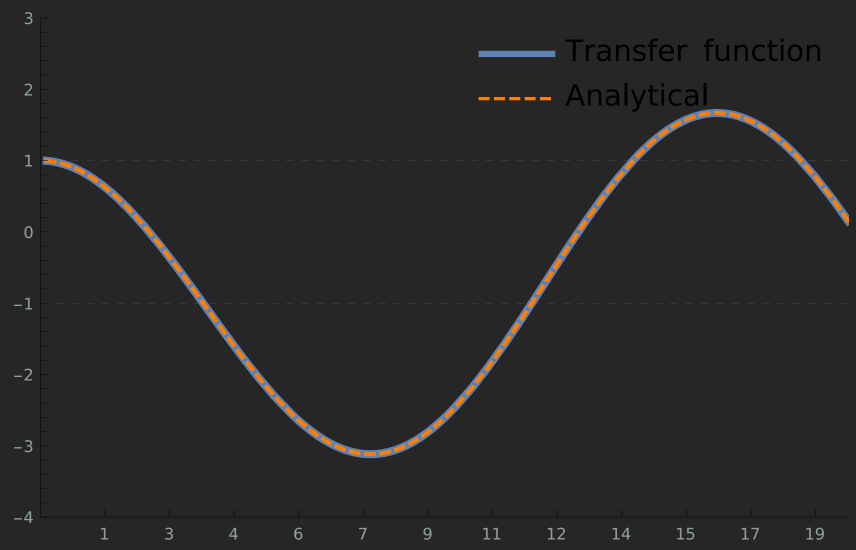
```
numberAddends = Binomial[NN, #] & /@ numberTerms ;
indexVectors = If[NN > #[[1]],
      Join[ConstantArray [1, #[[1]]], ConstantArray [0, NN - #[[1]]]],
      ConstantArray [1, #[[1]]]
    ] & /@ Transpose @{numberTerms , numberAddends };
indexSets = Flatten[Permutations [#] & /@ indexVectors , 1];
anTF = 1;
Do[
  layers = Flatten @Position[indexSets [[term]], 1];
  Zs = Sqrt[ρ# * μ#] & /@ layers ;
  sortedZs = Zs[[# ;;  ;; 2]] & /@ {1, 2};
  anTF = anTF + (-1)^(Length [layers ]/2) * 0.5 *
     (Times @@ sortedZs [[1]] / Times @@ sortedZs [[2]] + Times @@ sortedZs [[2]] / Times @@ sortedZs [[1]]) Times @@ (Tan[r#] & /@ layers);
  , {term , 1, Length @indexSets }];
anTF = (Times @@ (Cos[r#] & /@ Range[Length @hs])) * anTF ;
(*Prepare  to  ecaluate  numerically *)
auxVar = (anTF /. r1 -> (2 π * f * hs[[1]]) / (Vs[[1]] * Sqrt[1 + ⅈ * DE])) /. μ1 ρ1 -> (rhos[[1]] * Vs[[1]])^2;
Do[
  auxVar2 = auxVar ;
  auxVar = (auxVar2 /. ri -> (2 π * f * hs[[i]]) / (Vs[[i]] * Sqrt[1 + ⅈ * DE])) /. μi ρi -> (rhos[[i]] * Vs[[i]])^2;
  , {i, 2, NN}];
auxVar = Expand[auxVar];
(*Make  table  evaluating  numerically *)
myTF = Table[{flist[[k]], auxVar /. f -> flist[[k]]}, {k, 1, Length @flist}];
(*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%    *)
(*The  other  formula  ---------------------------------------------------------------------*)
nAddends = 2^(NN-1); (*different  coefficients  and  different  cosine  arguments *)
(*Possible  combinations  for  the  coefficient  --  impedance  constrats  --*)
Ccombos = {ConstantArray [0, NN]};
Do[
  Ccombos =
   Join[Ccombos , Permutations [Join[ConstantArray [1, 2 * ii], ConstantArray [0, NN - 2 ii]]]]
  , {ii , 1, Floor[NN / 2]}];
(*Combination  for  the  argument  of  the  cosines *)
Scombos = Join[{1}, #] & /@
   Permutations [Join[ConstantArray [1, NN - 1], ConstantArray [-1, NN - 1]], {NN - 1}];
cyclesMatrix = Table[Scombos [[kk]].Ccombos [[ii]], {ii , 1, nAddends }, {kk , 1, nAddends }];
```

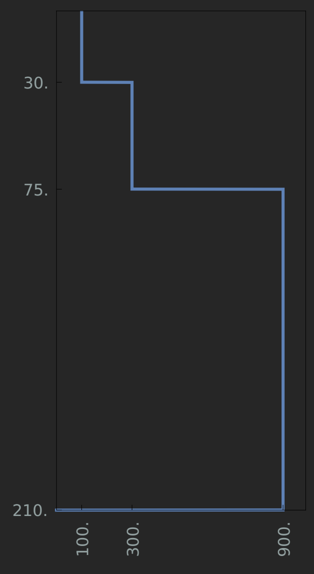
```
(*Create the coefficients to generate the impedance contrasts*)
impeExps = {};
Do[
  thisCombo = Ccombos[[ii]];
  If[Total[thisCombo] == 0, impeExps = AppendTo[impeExps, ConstantArray[0, NN]],
   AppendTo[impeExps, {1, 0, 0, 0}];
   auxFlag = 1;
   (*the first term is always 1*)
   Do[
    Which[
      thisCombo[[jj]] == 1 && auxFlag == 1, (impeExps[[-1]][[jj]] = -1;
       auxFlag = 0), (**)
      thisCombo[[jj]] == 1 && auxFlag == 0, (impeExps[[-1]][[jj]] = +1;
       auxFlag = 1),
      thisCombo[[jj]] == 0 && auxFlag == 0, impeExps[[-1]][[jj]] = 0,
      thisCombo[[jj]] == 0 && auxFlag == 1, impeExps[[-1]][[jj]] = 0
    ]
    , {jj, 2, NN}]
  ]
  , {ii, 1, Length@Ccombos}];
(*Create impedance contrasts... all of them!*)
impeLists = {1};
Do[
  layers = Flatten@Position[Ccombos[[term]], 1];
  Zs = Sqrt[ρ# * μ#] & /@ layers;
  sortedZs = Zs[[# ;; ;; 2]] & /@ {1, 2};
  AppendTo[impeLists,
   (-1)^(Length[layers]/2) * 1/2 (Times @@ sortedZs[[1]] / Times @@ sortedZs[[2]] + Times @@ sortedZs[[2]] / Times @@ sortedZs[[1]])];
  , {term, 2, Length@Ccombos}];
discreteSpectrum =
  Table[Total[Scombos[[ii]][[#]] * r# & /@ Range[NN]], {ii, 1, Length@Scombos}];
cosineList = Cos[#] & /@ discreteSpectrum;
amplitudes =
  Table[Total[Table[impeLists[[ii]] * (-1)^(cyclesMatrix[[ii, kk]]/2), {ii, 1, Length@impeLists}]],
   {kk, 1, Length@Scombos}];
anTFv2 = (amplitudes.cosineList) / 2^(NN-1);
(*Prepare to evaluate numerically... the amplitudes and the periods*)
amplitudesNum = amplitudes /. μ1 ρ1 → (rhos[[1]] * Vs[[1]])^2;
```

```
periodsNum = discreteSpectrum /. r1 → hs[[1]]/Vs[[1]];
Do[
  auxVar2 = amplitudesNum ;
  auxVar3 = periodsNum ;
  amplitudesNum = auxVar2 /. μi ρi → (rhos[[i]] * Vs[[i]])^2;
  periodsNum = auxVar3 /. ri → hs[[i]]/Vs[[i]];
  , {i, 2, NN}];
amplitudesNum = 1/2^(NN-1) * amplitudesNum ;
(*Prepare to evaluate numerically *)
auxVar = (anTFv2 /. r1 → (1 * f * hs[[1]])/(Vs[[1]] * Sqrt[1 + ⅈ * DE])) /. μ1 ρ1 → (rhos[[1]] * Vs[[1]])^2;
Do[
  auxVar2 = auxVar;
  auxVar = (auxVar2 /. ri → (1 * f * hs[[i]])/(Vs[[i]] * Sqrt[1 + ⅈ * DE])) /. μi ρi → (rhos[[i]] * Vs[[i]])^2;
  , {i, 2, NN}];
auxVar = Expand[auxVar];
(*Make table evaluating numerically *)
flist2 = Subdivide [0, fmax , 10 000];
myTFv2 = Table[{flist2 [[k]], auxVar /. f → flist2 [[k]]}, {k, 1, Length @ flist2}];
interpAnalyticFreq = Interpolation [myTFv2];
(*%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%*)
freqTicks = Table[{ii, ToString[Floor[ii * 10/(2 π)]]}, {ii, 1, fmax}];
(* 1/4((1+Z1+Z2*Z1+Z2)Cos[(r1+r2+r3)x]+(1-Z1+Z2*Z1-Z2)Cos[(r1-r2+r3)x]+
   (1+Z1-Z2*Z1-Z2)Cos[(r1+r2-r3)x]+(1-Z1-Z2*Z1+Z2)Cos[(r1-r2-r3)x]),*)
harmonicList [x_] := {1/4 ((1 + Z1 + Z2 * Z1 + Z2) Cos[(r1 + r2 + r3) x]),
   1/4 ((1 - Z1 + Z2 * Z1 - Z2) Cos[(r1 - r2 + r3) x]),
   1/4 ((1 + Z1 - Z2 * Z1 - Z2) Cos[(r1 + r2 - r3) x]),
   1/4 ((1 - Z1 - Z2 * Z1 + Z2) Cos[(r1 - r2 - r3) x])}
TFPlot = Plot[{
     interpExactExpFreq [x],
     interpAnalyticFreq [x]}
```

```
    , {x, fmin, fmax},
    PlotRange → {{0, fmax}, {-4, 3}},
    PlotStyle → {{Automatic, Thickness[0.01]}, {Orange, Dashed, Thickness[0.005]},
      {Black, Dotted}, {Black, Dotted}, {Black, Dotted}, {Black, Dotted}},
    Axes → False,
    GridLines → {None, {{1, Dashed}, {-1, Dashed}}},
    PlotStyle → {{Automatic, Thickness[0.012]},
      {Orange, Dashed, Thickness[0.015/3]}},
    Frame → {{True, False}, {True, False}},
    FrameTicks → {{Automatic, None}, {freqTicks, None}},
    PlotLegends →
     Placed[LineLegend[Style[#, FontSize → 16] & /@ {"Transfer  function", "Analytical"},
       LegendLayout → "Column"], {Right, Top}],
    RotateLabel -> False
   ];
(*Plot Evolution*)
ProfilePlot =
  ParametricPlot[{Piecewise[Table[{Vs[[ii]], z < Depths[[ii]]}, {ii, 1, Length[Vs]}]], -z},
   {z, 0, Depths[[-1]]},
   PlotRange → {{0, Max[Vs[[1 ;; -1]]] * 1.1}, {0, -Depths[[-1]]}},
   Frame → True,
   FrameTicks → {{Zticks, None}, {Vticks, None}},
   AspectRatio → 2];
AuxTableFigures = With[{size = 400},
  Row[Show[#, ImageSize → {Automatic, size}, ImagePadding → {{75, 15}, {90, 50}}] & /@
    {TFPlot, ProfilePlot}]]
```

Out[217]=


Transfer function
Analytical
3
2
1
0
−1
−2
−3
−4
1
3
4
6
7
9
11
12
14
15
17
19


30.
75.
210.
100.
300.
900.

In[237]:=
```
(*Plot Spectrum ------------------------------------------------------------*)
colorFun = Function[{x, y},
   Blend[{{Min[amplitudesNum], Yellow}, {Max[amplitudesNum], Red}}, Norm[{x, y}]]];
halfSpectrum = Transpose@{Abs[periodsNum], amplitudesNum};
spectraTableFigures =
  ListPlot[halfSpectrum,
   Axes → {True, False},
   Filling → Axis,
   Frame → True,
   PlotStyle → {Darker@Orange, PointSize[Large]},
   ColorFunction → colorFun,
   AspectRatio → 1];
```

In[240]:=
```
(*Plot Dispersion
  Relation ------------------------------------------------------------*)
dispFunction[x_] = ArcCos[interpAnalyticFreq[x]];
dispRel = Table[{ArcCos[Re@myTFv2[[kk]][[2]]], myTFv2[[kk]][[1]]}, {kk, 1, Length@myTFv2}];
drTableFigures =
  ParametricPlot[{ArcCos[interpAnalyticFreq[x]], x}, {x, 0., fmax},
   PlotRange → {{0, π}, {fmin, fmax}},
   PlotStyle → {Darker@Red, Thickness[0.015/1.5]},
   AspectRatio → 1.5,
   Axes → False,
   Frame → True,
   FrameTicks → {{freqTicks, None}, {{{0.05, 0}, {0.99 * π, π}}, None}}];
```

In[243]:=
```
(*Combine Plots*)
AuxTableFigures2 = With[{size = 400},
  Row[Show[#, ImageSize → {Automatic, size}, ImagePadding → {{75, 10}, {90, 50}}] & /@
    {drTableFigures, spectraTableFigures}]]
```

Out[243]=