Analytic Formulae of 1D transfer functions for layered sites

This notebook implements the calculations of exact transfer functions for layered soils, including the computation of Horizontal-to-Vertical Spectral Ratios, as described in Kawase *et al* (2011).

Preliminaries

Fix some parameters

Image style:

In[1]:=
```
texStyle = {FontSize → 14, FontFamily → "Times", FontColor → Black};
```

Frequency range to take into consideration:

In[2]:=
```
fmin = 0.01; fmax = 12;
```

Set work directory (wherever the txt files containing the stations info are):

```
directoryName = ""; (*Insert directory name here*)
SetDirectory[directoryName];
```

Set damping for the plots (since we are not calculating the actual damping in the site but choosing one for display purposes, let is fix the same value for all the sites):

In[5]:=
```
DE = 0.1;
```

Implementing the layer matrices

This list, when evaluated (see that as per definition the evaluation is delayed), generates the layer matrices for a given set of impedances (“alphas”), layer heights (“hs”) and shear-wave velocities (“Vs”).

The matrices relate the ground displacement to the displacement at the base of the last layer, the last one being the last one whose thickness is specified in the KikNet data.

In[6]:=
```
Llist := Table[
  {{1/2 (1 + alphas[[jj]]) * Exp[ⅈ * (ω * hs[[jj]])/Vs[[jj]]], 1/2 (1 - alphas[[jj]]) * Exp[-ⅈ * (ω * hs[[jj]])/Vs[[jj]]]},
       {1/2 (1 - alphas[[jj]]) * Exp[ⅈ * (ω * hs[[jj]])/Vs[[jj]]],
   1/2 (1 + alphas[[jj]]) * Exp[-ⅈ * (ω * hs[[jj]])/Vs[[jj]]]}}, {jj, 1, Length[alphas]}];
```

In[7]:=
```
LlistP := Table[
  {{1/2 (1 + alphasP[[jj]]) * Exp[ⅈ * (ω * hs[[jj]])/Vp[[jj]]], 1/2 (1 - alphasP[[jj]]) * Exp[-ⅈ * (ω * hs[[jj]])/Vp[[jj]]]},
       {1/2 (1 - alphasP[[jj]]) * Exp[ⅈ * (ω * hs[[jj]])/Vp[[jj]]],
   1/2 (1 + alphasP[[jj]]) * Exp[-ⅈ * (ω * hs[[jj]])/Vp[[jj]]]}}, {jj, 1, Length[alphas]}];
```

Data access

Data downloaded from https://www.kyoshin.bosai.go.jp/cgi-bin/kyoshin/db/sitedat.cgi?1+NM-RH04+kik ... etc.

The 10 sites to be considered in this study:

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

Example:

In[9]:=
```
Framed[Style[Import[SitesList[[9]]], FontSize → 14]]
```

Out[9]=

No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	4.00,	4.00,	330.00,	100.00
2,	4.00,	8.00,	330.00,	160.00
3,	12.00,	20.00,	1580.00,	160.00
4,	18.00,	38.00,	1580.00,	260.00
5,	60.00,	98.00,	1580.00,	290.00
6,	48.00,	146.00,	1580.00,	320.00
7,	40.00,	186.00,	1710.00,	370.00
8,	-----,	-----,	1710.00,	410.00

Writing the (reciprocal of) exact transfer functions:

Displacement-to-displacement, base-to-top (u_{top} / u_{base})

First, write it in product form (eq 10).

Pick a number of layers (NN) :

In[10]:=
```
NN = 3;
(*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) * (Times @@ sortedZs[[1]])/(Times @@ sortedZs[[2]]) Times @@ (Tan[r#] & /@ layers);
  , {term, 1, Length@indexSets}];
anTF = (Times @@ (Cos[r#] & /@ Range[NN])) * anTF;
Framed[anTF]
```

Out[19]=

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

Construct also the harmonic decomposition (eq 7):

In[20]:=
```
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) * (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[33]=

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

Surface-to-impinging-amplitude $(u_{top}\,/\,S_{inc})$

Pick a number of layers (NN) :

In[34]:=

```
NN = 3;
(*--------- The first addend does not change -----------------*)
(*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];
anTF1 = 1;
Do[
  layers = Flatten@Position[indexSets[[term]], 1];
  Zs = Sqrt[ρ# * μ#] & /@ layers;
  sortedZs = Zs[[# ;; ;; 2]] & /@ {1, 2};
  anTF1 =
   anTF1 + (-1)^(Length[layers]/2) * (Times @@ sortedZs[[1]])/(Times @@ sortedZs[[2]]) Times @@ (Tan[r#] & /@ layers);
  , {term, 1, Length@indexSets}];
(*--------- The first second (imaginary par) does not change -----------------*)
(*maximum number of tangents in any expression*)
maxNumberTerms = 1 + Floor[NN - 1, 2];
(*number of factors in each group of expressions*)
numberTerms = (1 + 2 #) & /@ Join[{0}, Range[maxNumberTerms/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];
anTF2 = 0;
Do[
  layers = Flatten@Position[indexSets[[term]], 1];
  Zs = Sqrt[ρ# * μ#] & /@ layers;
  sortedZs = Zs[[# ;; ;; 2]] & /@ {1, 2};
  anTF2 =
   anTF2 - (-1)^((Length[layers]-1)/2) * (Times @@ sortedZs[[1]])/(Times @@ sortedZs[[2]]) Times @@ (Tan[r#] & /@ layers);
  , {term, 1, Length@indexSets}];
anTF = (Times @@ (Cos[r#] & /@ Range[NN])) * (anTF1 - anTF2 * ⅈ/Sqrt[ρhalf * μhalf]);
Framed[anTF]
```

Out[50]=

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

Using analytical formula for second transfer function

Frequency range to take into consideration:

In[51]:=

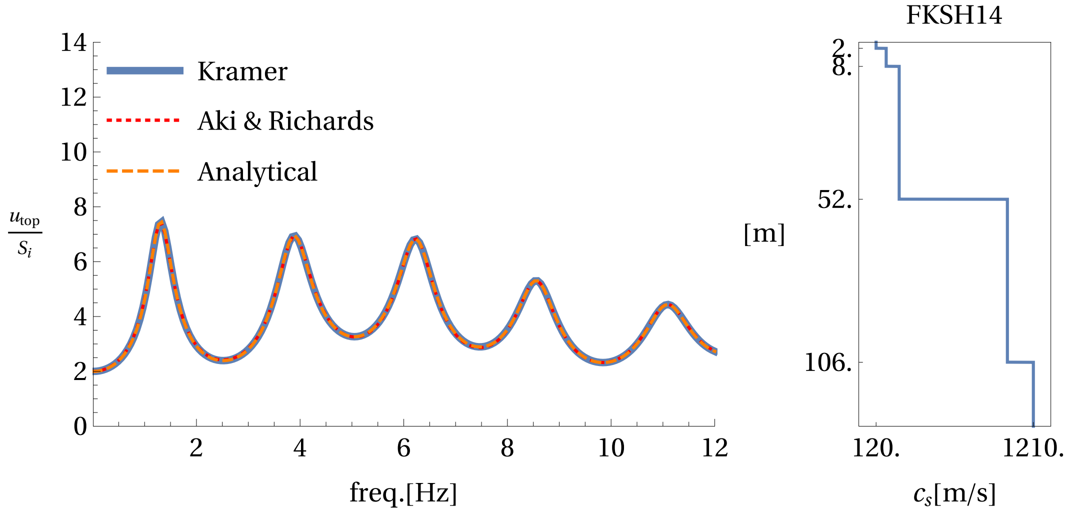
```
DE = 0.05;
AuxTableFigures = Table[{}, {Length[SitesList]}];
Quiet@Do[(*Print[jj];*)
   (*Import and Prepare
     Data -------------------------------------------------------------*)
   data = Import[SitesList[[jj]], "Table"];
   SiteName = StringSplit[ToUpperCase[SitesList[[jj]]], "."][[1]];
   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}];
(*Bedrock Properties*)
Vhalf = Vs[[-1]];
rhohalf = 1500;
muhalf = rhohalf * Vhalf^2;
(*Frequency interval*)
flist = Subdivide[fmin, fmax, 200];
(*Kramer's*)
L[ω_] = Llist[[1]];
For[ii = 2, ii ≤ Length[alphas], ii++, L[ω_] = Llist[[ii]].L[ω]];
TF[ω_] = 2 / (L[ω][[1, 1]] + L[ω][[1, 2]]);
TFKramer =
 Table[{flist[[kk]], Abs[TF[(2 π) / Sqrt[1 + ⅈ * DE] flist[[kk]]]]}, {kk, 1, Length@flist}];
(*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 ,
```

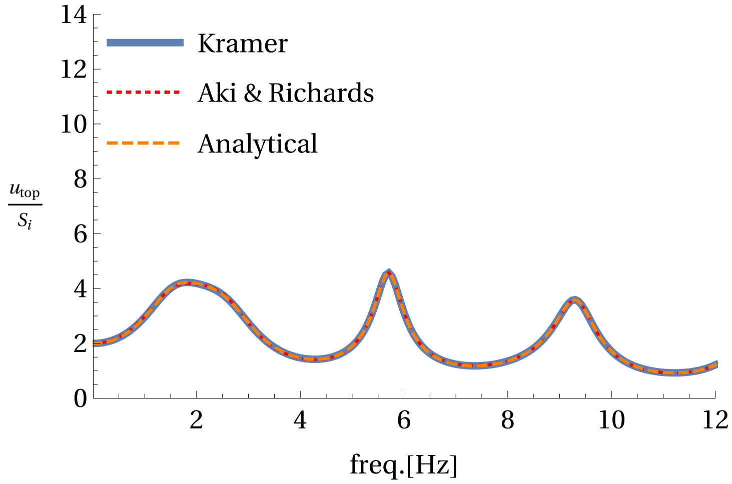
```
    {flist[[ii]], 2 / Abs[exactExp[[1, 1]] - ⅈ * exactExp[[2,1]] / (2 π*flist[[ii]]*Sqrt[muhalf*rhohalf])]}]
  , {ii, 1, Length@flist}];
(*Compute and Evaluate Analytical ---------
 --------------------------------------------------*)
NN = Length@hs;
(*--------- The first addend ------------------*)
(*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];
anTF1 = 1;
Do[
  layers = Flatten@Position[indexSets[[term]], 1];
  Zs = Sqrt[ρ# * μ#] &/@ layers;
  sortedZs = Zs[[# ;;  ;; 2]] &/@ {1, 2};
  anTF1 =
   anTF1 + (-1)^(Length[layers]/2) * (Times @@ sortedZs[[1]]) / (Times @@ sortedZs[[2]]) Times @@ (Tan[r#] &/@ layers);
  , {term, 1, Length@indexSets}];
(*--------- The second addend (imaginary part) ------------------*)
(*maximum number of tangents in any expression*)
maxNumberTerms = 1 + Floor[NN - 1, 2];
(*number of factors in each group of expressions*)
numberTerms = (1 + 2 #) &/@ Join[{0}, Range[maxNumberTerms / 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];
anTF2 = 0;
Do[
  layers = Flatten@Position[indexSets[[term]], 1];
```

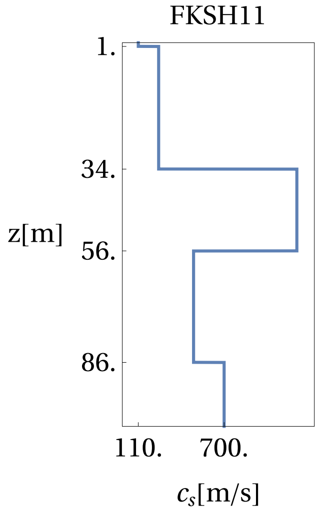
```
  Zs = Sqrt[ρ# * μ#] & /@ layers;
  sortedZs = Zs[[# ;;  ;; 2]] & /@ {1, 2};
  anTF2 =
   anTF2 - (-1)^((Length[layers]-1)/2) * (Times @@ sortedZs[[1]])/(Times @@ sortedZs[[2]]) Times @@ (Tan[r#] & /@ layers);
  , {term, 1, Length @ indexSets}];
anTF = (Times @@ (Cos[r#] & /@ Range[NN])) * (anTF1 - anTF2 * ⅈ/Sqrt[ρhalf * μhalf]);
(*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) /.
   μhalf ρhalf → (rhohalf * Vhalf)^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]], 2/Abs[auxVar /. f → flist[[k]]]}, {k, 1, Length @ flist}];
(*Plot Transfer
   function ==========================================================*)
TFPlot = ListLinePlot[{
    TFKramer,
    exactExpFreq,
    myTF},
   PlotRange → {{fmin, fmax}, {0, Ceiling[1.1 * Max[Join[exactExpFreq, myTF]]]}},
   Axes → False,
   PlotStyle → {{Automatic, Thickness[0.012]},
     {Red, Dotted, Thickness[0.015/3]}, {Orange, Dashed, Thickness[0.015/3]}},
   Frame → {{True, False}, {True, False}},
   FrameLabel → {{"utop/Si", None}, {"freq.[Hz]", None}},
   PlotLegends → Placed[LineLegend[{"Kramer", "Aki & Richards", "Analytical"},
      LegendLayout → "Column"], {0.24, Top}],
   RotateLabel -> False,
   LabelStyle → texStyle];
(*Plot Evolution ==================*)
Vticks = {Vs[[1]], Vs[[-1]]};
Zticks = Table[{-Depths[[ii]], Depths[[ii]]}, {ii, Length[Depths]}];
ProfilePlot = ParametricPlot[
```

```
        {Piecewise[Join[Table[{Vs[[ii]], z < Depths[[ii]]}, {ii, 1, Length[Vs] - 1}],
            {{Vs[[-1]], 1.3 * Depths[[-1]] > z ≥ Depths[[-1]]}}]], -z}, {z, 0, 1.2 * Depths[[-1]]},
        PlotRange → {{0, Max[Vs[[1 ;; -1]]] * 1.1}, {0, -1.2 * Depths[[-1]]}},
        Frame → True,
        FrameLabel → {{Rotate["z[m]", -90 Degree], None}, {"cs[m/s]", SiteName}},
        FrameTicks → {{Zticks, None}, {Vticks, None}},
        LabelStyle → texStyle,
        AspectRatio → 2,
        PlotRangeClipping → True];
    (*Combine Plots*)
    AuxTableFigures[[jj]] =
     Row[Show[#, ImageSize → {Automatic, 300}, ImagePadding → {{60, 15}, {70, 30}}] & /@
        {TFPlot, ProfilePlot}];
    , {jj, 1, Length@SitesList}];
Column[{AuxTableFigures[[#]], Framed[Import[SitesList[[#]]]]} & /@ Range[10]]
```



No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	2.00,	2.00,	500.00,	120.00
2,	6.00,	8.00,	1410.00,	190.00
3,	44.00,	52.00,	1410.00,	280.00
4,	54.00,	106.00,	2210.00,	1030.00
5,	-----,	-----,	2740.00,	1210.00

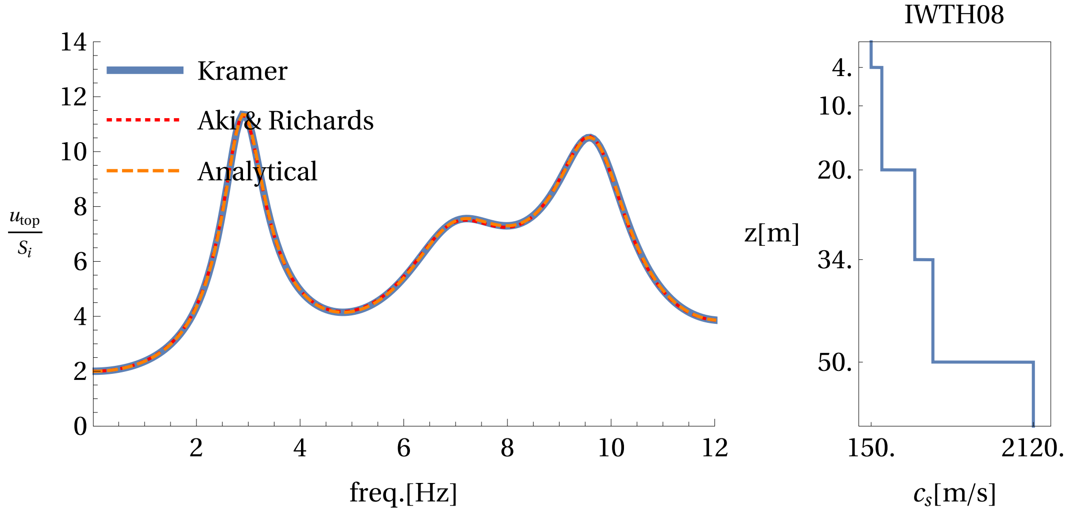
{ 



,

No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	1.00,	1.00,	200.00,	110.00
2,	33.00,	34.00,	1600.00,	250.00
3,	22.00,	56.00,	2200.00,	1200.00
4,	30.00,	86.00,	1700.00,	490.00
5,	-----,	-----,	1900.00,	700.00

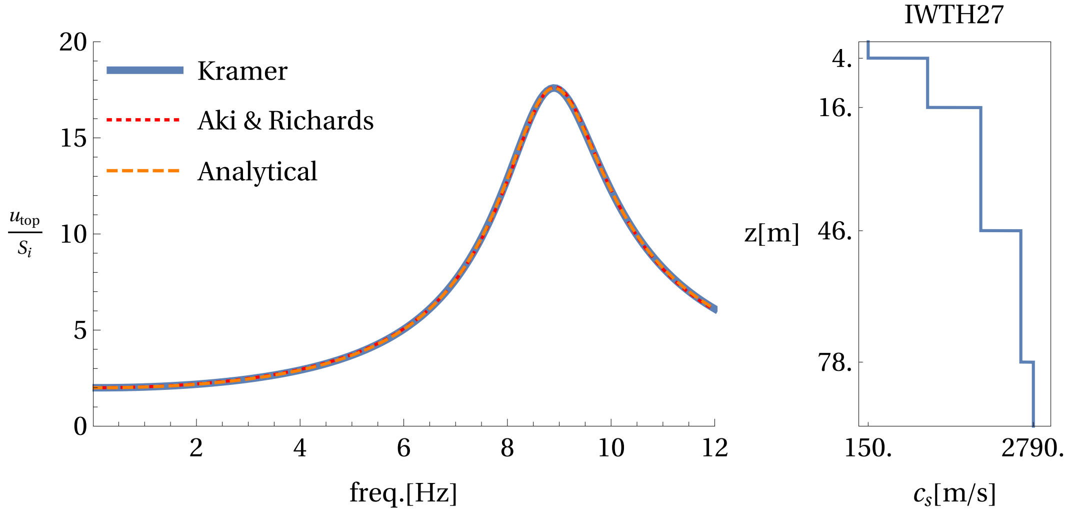
}

{
,

No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	4.00,	4.00,	360.00,	150.00
2,	6.00,	10.00,	600.00,	280.00
3,	10.00,	20.00,	2150.00,	280.00
4,	14.00,	34.00,	3000.00,	680.00
5,	16.00,	50.00,	3000.00,	900.00
6,	-----,	-----,	3680.00,	2120.00

}

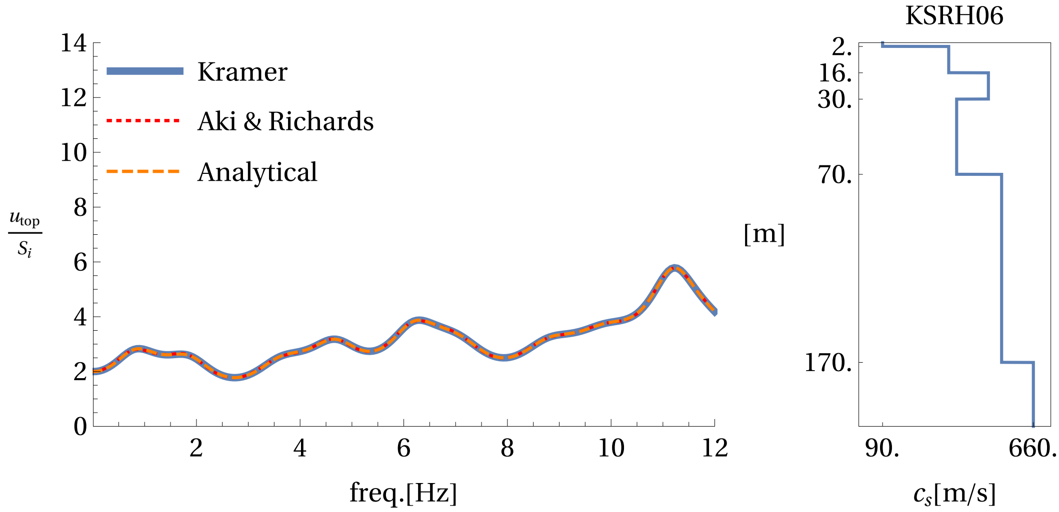
{



,

No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	4.00,	4.00,	360.00,	150.00
2,	12.00,	16.00,	4320.00,	1100.00
3,	30.00,	46.00,	4320.00,	1950.00
4,	32.00,	78.00,	5250.00,	2590.00
5,	-----,	-----,	5250.00,	2790.00

}

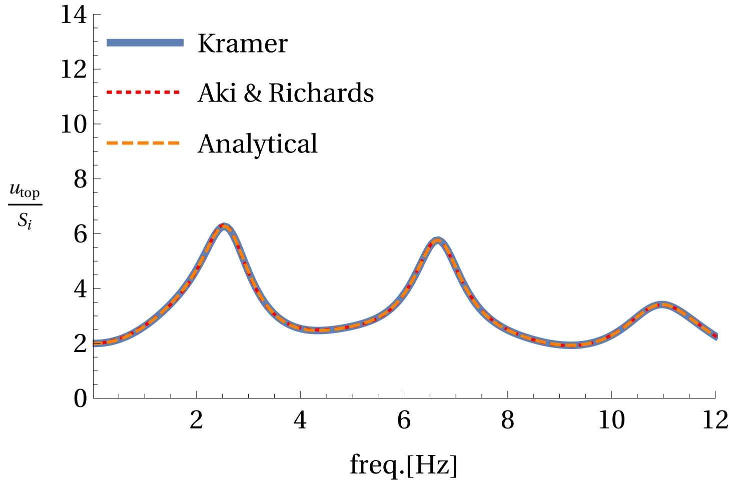
{

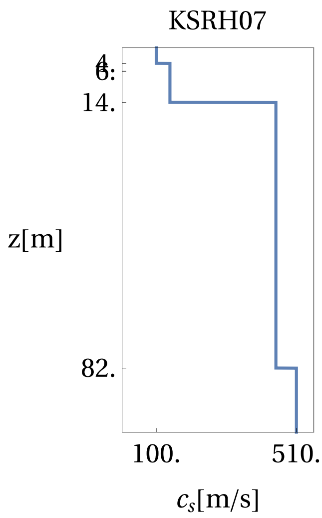
,

No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	2.00,	2.00,	180.00,	90.00
2,	14.00,	16.00,	940.00,	340.00
3,	14.00,	30.00,	1620.00,	490.00
4,	40.00,	70.00,	1620.00,	370.00
5,	100.00,	170.00,	1620.00,	540.00
6,	-----,	-----,	1950.00,	660.00

}

Out[54]=

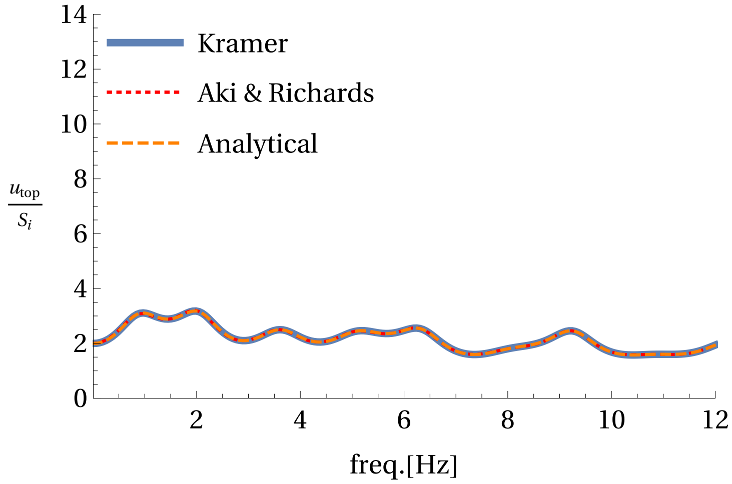
$\left\{\right.$ 

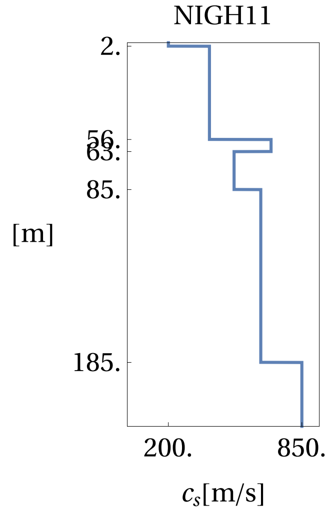


,

No	Thickness	Depth	Vp	Vs
	(m)	(m)	(m/s)	(m/s)
1,	4.00,	4.00,	330.00,	100.00
2,	2.00,	6.00,	330.00,	140.00
3,	8.00,	14.00,	1590.00,	140.00
4,	68.00,	82.00,	1590.00,	450.00
5,	-----,	-----,	1790.00,	510.00

$\left.\right\}$

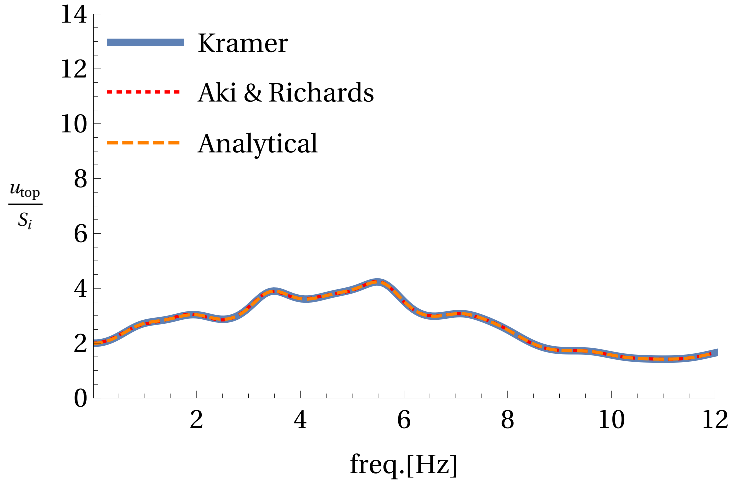
{ 

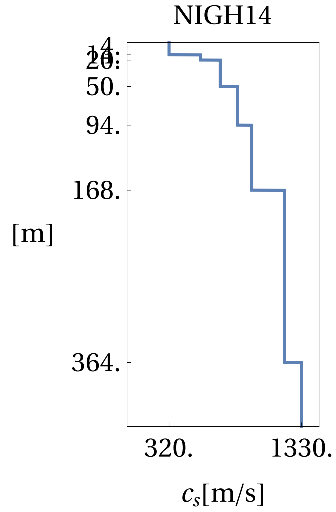


,

No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	2.00,	2.00,	500.00,	200.00
2,	54.00,	56.00,	1830.00,	400.00
3,	7.00,	63.00,	1830.00,	700.00
4,	22.00,	85.00,	1830.00,	520.00
5,	100.00,	185.00,	1830.00,	650.00
6,	-----,	-----,	2080.00,	850.00

}

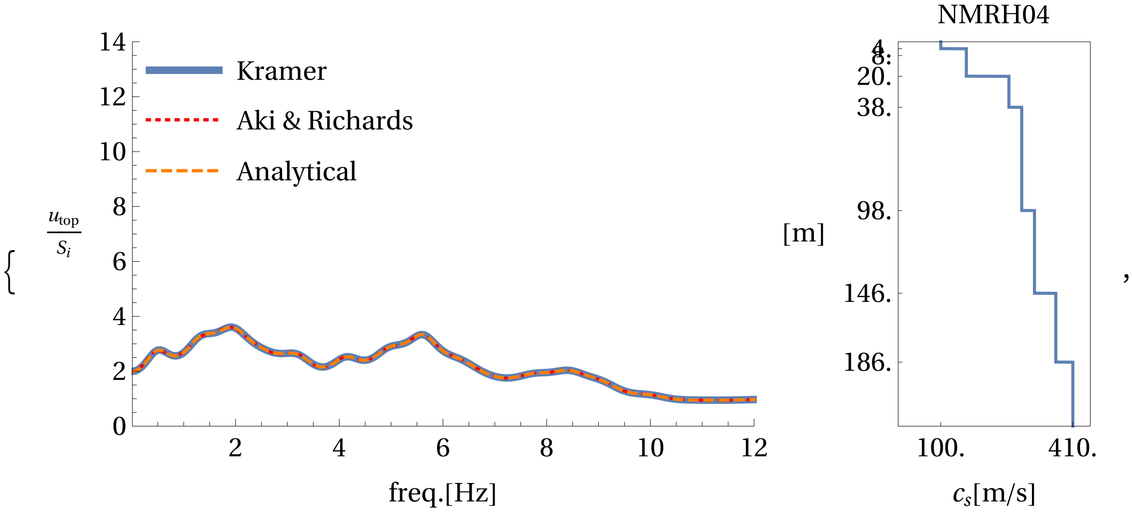
{ 



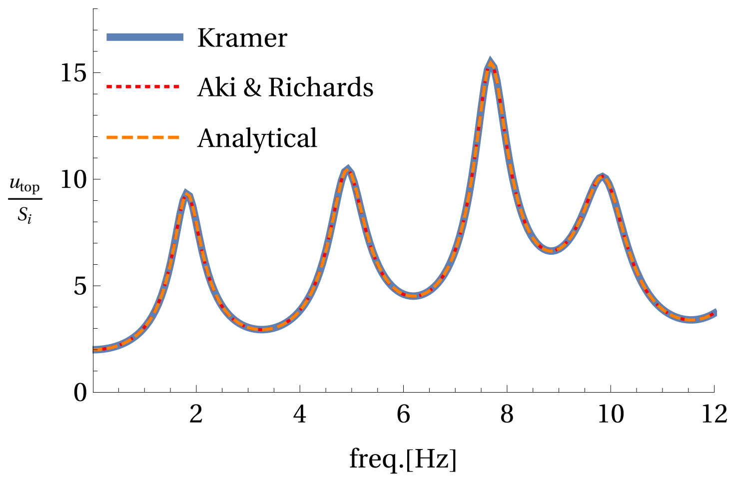
,

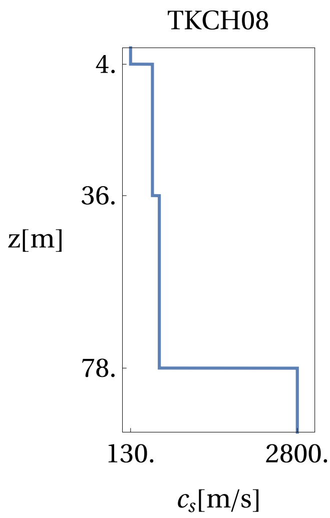
No	Thickness	Depth	Vp	Vs
	(m)	(m)	(m/s)	(m/s)
1,	4.00,	4.00,	550.00,	320.00
2,	10.00,	14.00,	1850.00,	320.00
3,	6.00,	20.00,	1850.00,	560.00
4,	30.00,	50.00,	1850.00,	710.00
5,	44.00,	94.00,	2250.00,	840.00
6,	74.00,	168.00,	2530.00,	950.00
7,	196.00,	364.00,	2760.00,	1200.00
8,	-----,	-----,	3020.00,	1330.00

}



No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	4.00,	4.00,	330.00,	100.00
2,	4.00,	8.00,	330.00,	160.00
3,	12.00,	20.00,	1580.00,	160.00
4,	18.00,	38.00,	1580.00,	260.00
5,	60.00,	98.00,	1580.00,	290.00
6,	48.00,	146.00,	1580.00,	320.00
7,	40.00,	186.00,	1710.00,	370.00
8,	-----,	-----,	1710.00,	410.00





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

HV ratio

Borrowing this function from https://mathematica.stackexchange.com/questions/13226/how-can-i-get-exactly-5-logarithmic-divisions-of-an-interval.

In[55]:=
```
logspace[increments_, start_?Positive, end_?Positive] :=
 Exp@Range[Log@start, Log@end, Log[end/start]/increments]
```

In[56]:=
```
(*---------------------------------------------------------------------------
-----------------------------------------
----*)
(*Frequency interval*)
fmin = 0.01; fmax = 100;
flist = logspace[1000, fmin, fmax];
```

In[63]:=
```
DE = 0.1;
AuxTableFigures = Table[{}, {Length[SitesList]}];
insets = Table[{}, {Length[SitesList]}];
Quiet@Do[(*Print[jj];*)
  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}]];
Vp = Flatten[Table[data⟦ii, 4⟧, {ii, 3, Nrows}]];
Vp = ToExpression[StringDelete[#, ","]] & /@ Vp;
(*-------------------------------------------------------*)
Vbase = Vs[[-2]];
Vpbase = Vp[[-2]];
rhos = Table[1500., {ii, 1, Length[Vs]}];
(*Elastic constants*)
mus = Table[rhos[[ii]] * Vs[[ii]]^2, {ii, 1, Length[Vs]}];
lambdaDosmus = Table[rhos[[ii]] * Vp[[ii]]^2, {ii, 1, Length[Vp]}];
alphas = Table[(rhos[[ii]] * Vs[[ii]])/(rhos[[ii + 1]] * Vs[[ii + 1]]), {ii, 1, Nrows - 3}];
alphasP = Table[(rhos[[ii]] * Vp[[ii]])/(rhos[[ii + 1]] * Vp[[ii + 1]]), {ii, 1, Nrows - 3}];
(*Bedrock Properties*)
Vhalf = Vs[[-1]];
Vphalf = Vp[[-1]];
rhohalf = 1500;
muhalf = rhohalf * Vhalf^2;
lambdaDosmuhalf = rhohalf * Vphalf^2;
nus = (2 - (Vp[[#]] / Vs[[#]])^2)/(2 (1 - (Vp[[#]] / Vs[[#]])^2)) & /@ Range[Length@Vp];
nuhalf = nus[[-1]];
(*------------ FOR THE CURVATURE ---------------------------*)
tp = Table[hs[[ii]]/Vp[[ii]], {ii, 1, Length@hs}];
ts = Table[hs[[ii]]/Vs[[ii]], {ii, 1, Length@hs}];
γs = Total[Table[(rhos[[ii]] * Vs[[ii]])/(rhohalf * Vhalf) * ts[[ii]], {ii, 1, Length@hs}]];
γp = Total[Table[(rhos[[ii]] * Vp[[ii]])/(rhohalf * Vphalf) * tp[[ii]], {ii, 1, Length@hs}]];
```

```
κs = ts.ts + 2 * Total @ Flatten[Table[Table[ts[[ii]] * ts[[jj]] * (rhos[[ii]] * Vs[[ii]])/(rhos[[jj]] * Vs[[jj]]),
        {jj, ii + 1, Length @ hs}], {ii, 1, -1 + Length @ hs}]];
κp = tp.tp + 2 * Total @ Flatten[Table[Table[tp[[ii]] * tp[[jj]] * (rhos[[ii]] * Vp[[ii]])/(rhos[[jj]] * Vp[[jj]]),
        {jj, ii + 1, Length @ hs}], {ii, 1, -1 + Length @ hs}]];
(*----------------------------------------P WAVE--------
 ------------------------------------------------*)
layerMatrix = hs[[#]] * {{0., 1/(lambdaDosmus [[#]] * (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 = 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*)
 exactExpFreqP = AppendTo[exactExpFreq ,
    {flist[[ii]], 2/Abs[exactExp [[1, 1]] - ⅈ * exactExp [[2,1]]/(2 π*flist[[ii]]*Sqrt[lambdaDosmuhalf *rhohalf ])]}]
 , {ii, 1, Length @ flist}];
(*Compute and Evaluate Analytical ---------
 --------------------------------------------------*)
NN = Length @ hs;
(*--------- The first addend ------------------*)
(*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];
```

```
anTF1 = 1;
Do[
 layers = Flatten@Position[indexSets[[term]], 1];
 Zs = Sqrt[ρ# * μ#] & /@ layers;
 sortedZs = Zs[[# ;; ;; 2]] & /@ {1, 2};
 anTF1 =
  anTF1 + (-1)^(Length[layers]/2) * (Times @@ sortedZs[[1]])/(Times @@ sortedZs[[2]]) Times @@ (Tan[r#] & /@ layers);
 , {term, 1, Length@indexSets}];
(*--------- The second addend (imaginary part) -----------------*)
(*maximum number of tangents in any expression*)
maxNumberTerms = 1 + Floor[NN - 1, 2];
(*number of factors in each group of expressions*)
numberTerms = (1 + 2 #) & /@ Join[{0}, Range[maxNumberTerms/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];
anTF2 = 0;
Do[
 layers = Flatten@Position[indexSets[[term]], 1];
 Zs = Sqrt[ρ# * μ#] & /@ layers;
 sortedZs = Zs[[# ;; ;; 2]] & /@ {1, 2};
 anTF2 =
  anTF2 - (-1)^((Length[layers]-1)/2) * (Times @@ sortedZs[[1]])/(Times @@ sortedZs[[2]]) Times @@ (Tan[r#] & /@ layers);
 , {term, 1, Length@indexSets}];
anTF = (Times @@ (Cos[r#] & /@ Range[NN])) * (anTF1 - anTF2 * ⅈ/Sqrt[ρhalf * μhalf]);
(*Prepare to ecaluate numerically*)
auxVar = (((anTF /. r1 → (2 π * f * hs[[1]])/(Vp[[1]] * Sqrt[1 + ⅈ * DE])) /. μ1 ρ1 → (rhos[[1]] * Vs[[1]])^2) /.
   μhalf ρhalf → (rhohalf * Vphalf)^2;
Do[
 auxVar2 = auxVar;
 auxVar = (auxVar2 /. ri → (2 π * f * hs[[i]])/(Vp[[i]] * Sqrt[1 + ⅈ * DE])) /. μi ρi → (rhos[[i]] * Vp[[i]])^2;
 , {i, 2, NN}];
auxVar = Expand[auxVar];
```

```
(*Make table evaluating numerically*)
myTFp = Table[{flist[[k]], 2/(Abs[auxVar /. f → flist[[k]]])}, {k, 1, Length@flist}];
(*--------------------------------------S
 WAVE-----------------------------------------------------*)
(*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]], 2/(Abs[exactExp[[1, 1]] - ⅈ * exactExp[[2,1]]/(2 π*flist[[ii]]*Sqrt[muhalf*rhohalf])])}]
 , {ii, 1, Length@flist}];
(*Compute and Evaluate Analytical ---------
 --------------------------------------------------*)
NN = Length@hs;
(*--------- The first addend -----------------*)
(*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];
```

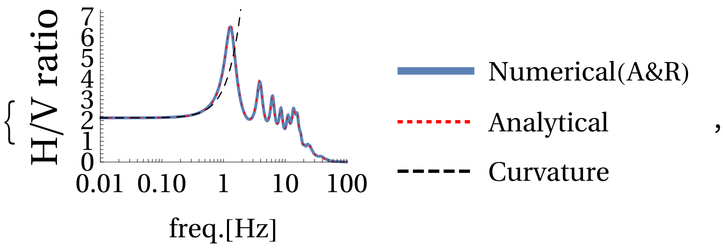
```
anTF1 = 1;
Do[
 layers = Flatten@Position[indexSets[[term]], 1];
 Zs = Sqrt[ρ# * μ#] & /@ layers;
 sortedZs = Zs[[# ;; ;; 2]] & /@ {1, 2};
 anTF1 =
  anTF1 + (-1)^(Length[layers]/2) * (Times @@ sortedZs[[1]])/(Times @@ sortedZs[[2]]) Times @@ (Tan[r#] & /@ layers);
 , {term, 1, Length@indexSets}];
(*--------- The second addend (imaginary part) -----------------*)
(*maximum number of tangents in any expression*)
maxNumberTerms = 1 + Floor[NN - 1, 2];
(*number of factors in each group of expressions*)
numberTerms = (1 + 2 #) & /@ Join[{0}, Range[maxNumberTerms/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];
anTF2 = 0;
Do[
 layers = Flatten@Position[indexSets[[term]], 1];
 Zs = Sqrt[ρ# * μ#] & /@ layers;
 sortedZs = Zs[[# ;; ;; 2]] & /@ {1, 2};
 anTF2 =
  anTF2 - (-1)^((Length[layers]-1)/2) * (Times @@ sortedZs[[1]])/(Times @@ sortedZs[[2]]) Times @@ (Tan[r#] & /@ layers);
 , {term, 1, Length@indexSets}];
anTF = (Times @@ (Cos[r#] & /@ Range[NN])) * (anTF1 - anTF2 * ⅈ/Sqrt[ρhalf * μhalf]);
(*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) /.
  μhalf ρhalf → (rhohalf * Vhalf)^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]], 2/Abs[auxVar /. f → flist[[k]]]}, {k, 1, Length@flist}];
(*------------------------------------ COMPUTE
 RATIOS -------------------------------------------------*)
myHVratio = Table[{myTF[[ii]][[1]], ((8 (1 - nuhalf))/(1 - 2 * nuhalf))^(1/4) * myTF[[ii]][[2]]/myTFp[[ii]][[2]]},
  {ii, 1, Length@myTFp}];
HVratio = Table[{exactExpFreq[[ii]][[1]], ((8 (1 - nuhalf))/(1 - 2 * nuhalf))^(1/4) * exactExpFreq[[ii]][[2]]/exactExpFreqP[[ii]][[2]]},
  {ii, 1, Length@exactExpFreq}];
approx = Table[{flist[[ii]], ((8 (1 - nuhalf))/(1 - 2 * nuhalf))^(1/4) *
    (1 + ((γp^2 - κp) - (γs^2 - κs))/2 * (2 π * flist[[ii]])^2)}, {ii, 1, Length@flist}];
inset = ListLogLinearPlot[{
    HVratio,
    myHVratio,
    approx
  },
  Joined → True,
  FrameTicks → {None, {{0.1, 1., 10.}, None}},
  PlotRange → {{0.1, 5.},
    {0.8 * ((8 (1 - nuhalf))/(1 - 2 * nuhalf))^(1/4), 1.2 * ((8 (1 - nuhalf))/(1 - 2 * nuhalf))^(1/4) * (1 + ((γp^2 - κp) - (γs^2 - κs))/2 * (2 π)^2)}},
  Axes → False,
  FrameTicksStyle → {Black, 12},
  PlotStyle → {{Automatic, Thickness[0.012]},
    {Red, Dotted, Thickness[0.015/3]}, {Black, Dashed, Thickness[0.015/3]}},
  Frame → True,
  PlotRangeClipping → True,
  ImageSize → Tiny,
  AspectRatio → 1/2];
insets[[jj]] = inset;
AuxTableFigures[[jj]] = ListLogLinearPlot[{
    HVratio,
    myHVratio,
    approx
  },
  Joined → True,
  PlotRange → {{fmin, fmax}, {0, 1.1 Max[myHVratio[[All, 2]]]}},
```

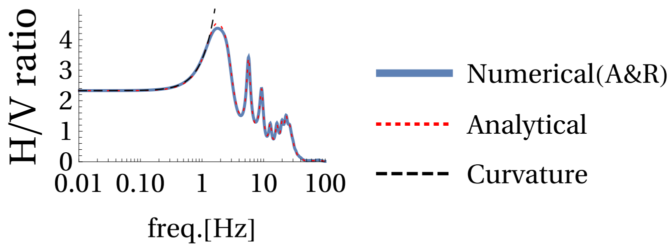
```
    Axes → False,
    PlotStyle → {{Automatic, Thickness[0.012]},
      {Red, Dotted, Thickness[0.015/3]}, {Black, Dashed, Thickness[0.015/3]}},
    Frame → {{True, False}, {True, False}},
    FrameLabel → {{Style["H/V ratio", 20], None}, {"freq.[Hz]", None}},
    PlotLegends → LineLegend[
      {"Numerical(A&R)", "Analytical", "Curvature"}, LegendLayout → "Column"],
    RotateLabel → True,
    LabelStyle → texStyle]
  , {jj, 1, Length@SitesList}]
```

In[67]:=

```
Column[{AuxTableFigures[[#]], Framed[Import[SitesList[[#]]]]} & /@ Range[10]]
```



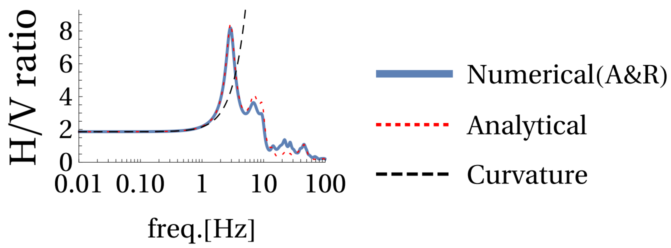
No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	2.00,	2.00,	500.00,	120.00
2,	6.00,	8.00,	1410.00,	190.00
3,	44.00,	52.00,	1410.00,	280.00
4,	54.00,	106.00,	2210.00,	1030.00
5,	-----,	-----,	2740.00,	1210.00

{ 

,

No	Thickness	Depth	Vp	Vs
	(m)	(m)	(m/s)	(m/s)
1,	1.00,	1.00,	200.00,	110.00
2,	33.00,	34.00,	1600.00,	250.00
3,	22.00,	56.00,	2200.00,	1200.00
4,	30.00,	86.00,	1700.00,	490.00
5,	-----,	-----,	1900.00,	700.00

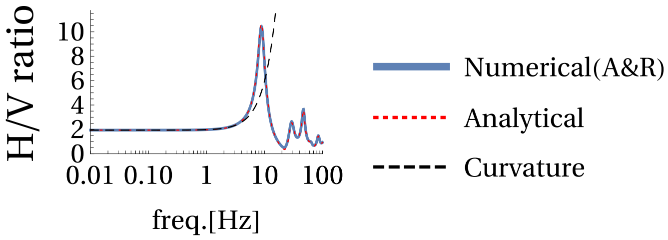
}

{ 

,

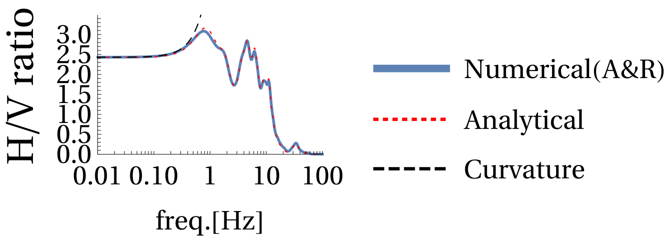
No	Thickness	Depth	Vp	Vs
	(m)	(m)	(m/s)	(m/s)
1,	4.00,	4.00,	360.00,	150.00
2,	6.00,	10.00,	600.00,	280.00
3,	10.00,	20.00,	2150.00,	280.00
4,	14.00,	34.00,	3000.00,	680.00
5,	16.00,	50.00,	3000.00,	900.00
6,	-----,	-----,	3680.00,	2120.00

}



,

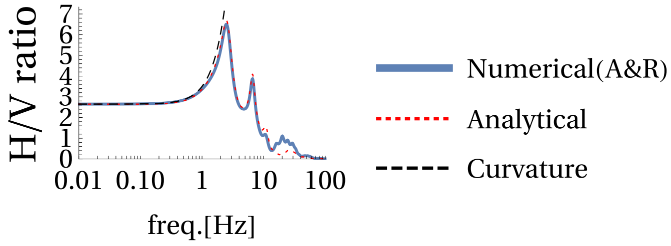
No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	4.00,	4.00,	360.00,	150.00
2,	12.00,	16.00,	4320.00,	1100.00
3,	30.00,	46.00,	4320.00,	1950.00
4,	32.00,	78.00,	5250.00,	2590.00
5,	-----,	-----,	5250.00,	2790.00



,

No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	2.00,	2.00,	180.00,	90.00
2,	14.00,	16.00,	940.00,	340.00
3,	14.00,	30.00,	1620.00,	490.00
4,	40.00,	70.00,	1620.00,	370.00
5,	100.00,	170.00,	1620.00,	540.00
6,	-----,	-----,	1950.00,	660.00

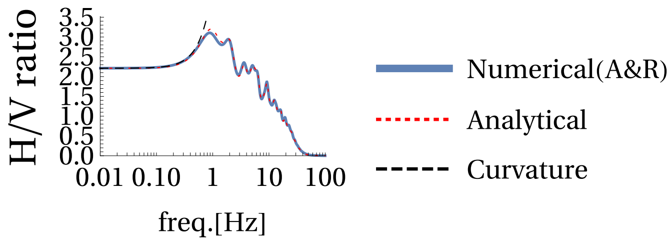
Out[67]=

{ 

,

No	Thickness	Depth	Vp	Vs
	(m)	(m)	(m/s)	(m/s)
1,	4.00,	4.00,	330.00,	100.00
2,	2.00,	6.00,	330.00,	140.00
3,	8.00,	14.00,	1590.00,	140.00
4,	68.00,	82.00,	1590.00,	450.00
5,	-----,	-----,	1790.00,	510.00

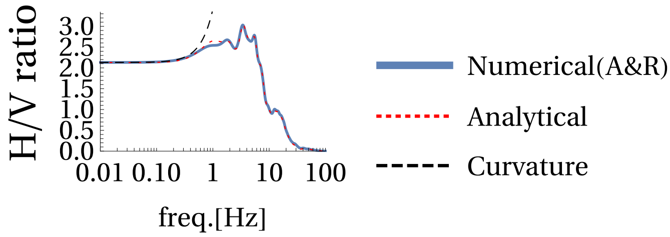
}

{ 

,

No	Thickness	Depth	Vp	Vs
	(m)	(m)	(m/s)	(m/s)
1,	2.00,	2.00,	500.00,	200.00
2,	54.00,	56.00,	1830.00,	400.00
3,	7.00,	63.00,	1830.00,	700.00
4,	22.00,	85.00,	1830.00,	520.00
5,	100.00,	185.00,	1830.00,	650.00
6,	-----,	-----,	2080.00,	850.00

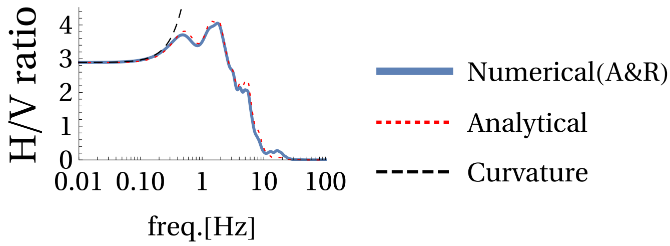
}

{ 

,

No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	4.00,	4.00,	550.00,	320.00
2,	10.00,	14.00,	1850.00,	320.00
3,	6.00,	20.00,	1850.00,	560.00
4,	30.00,	50.00,	1850.00,	710.00
5,	44.00,	94.00,	2250.00,	840.00
6,	74.00,	168.00,	2530.00,	950.00
7,	196.00,	364.00,	2760.00,	1200.00
8,	-----,	-----,	3020.00,	1330.00

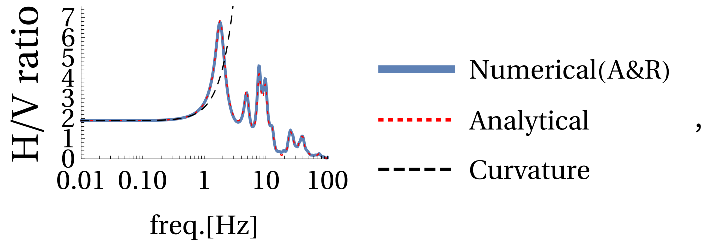
}

{ 

,

No	Thickness (m)	Depth (m)	Vp (m/s)	Vs (m/s)
1,	4.00,	4.00,	330.00,	100.00
2,	4.00,	8.00,	330.00,	160.00
3,	12.00,	20.00,	1580.00,	160.00
4,	18.00,	38.00,	1580.00,	260.00
5,	60.00,	98.00,	1580.00,	290.00
6,	48.00,	146.00,	1580.00,	320.00
7,	40.00,	186.00,	1710.00,	370.00
8,	-----,	-----,	1710.00,	410.00

}

{ 

,

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

}