Solutions for each layer of the shell. For the purposes of this worksheet, allow space for up to 20 layers.

```
v = Table[c1[k] * r^(n) + c2[k] * r^(-(n + 1)), {k, 1, 20}];

dv = D[v, r] + v / r;

Bn = dv[[1]] / u[1] /. r → R
```

$$\frac{n\,R^{-1+n}\,c1[1] + (-1-n)\,R^{-2-n}\,c2[1] + \frac{R^{n}\,c1[1]+R^{-1-n}\,c2[1]}{R}}{u[1]}$$

Define a function to get B.n just inside the inner boundary of the ABC region; this is called as part of the a minimization that computes permeability.

```
GetBn[enn_, dee_, seed_] := Module[{qsub, csub, eqs, m, k, intf, vars},
  m = Length[seed];
  qsub = Table[u[k] → seed[[k]], {k, 1, m}];
  qsub = Join[qsub, {R → 1, n → enn, d → dee}];
  eqs = {(v[[1]] /. r → R /. qsub) == 1, (v[[m]] /. r → R + m * d /. qsub) == 0};
  (* edges of boundary region *)
  For[k = 1, k < m, k++, (* interface conditions between layers in the boundary region *)
   intf = {(dv[[k]] / u[k] /. r → R + k * d /. qsub) ==
       (dv[[k + 1]] / u[k + 1] /. r → R + k * d /. qsub),
      (v[[k]] /. r → R + k * d /. qsub) == (v[[k + 1]] /. r → R + k * d /. qsub)};
   eqs = Join[eqs, intf];
  ];
  vars = Join[Table[c1[k], {k, 1, m}], Table[c2[k], {k, 1, m}]];
  csub = Solve[eqs, vars][[1]];
  Bn /. qsub /. csub
 ]
```

Now, figure out the permeabilities that satisfy the first m asymptotic boundary conditions. Do this via an error minimization. Note that working precision, accuracy, etc., have to be crazy, becuase the calculation isn't well conditioned.

```
GetPerm[een_, guess_] := Module[{ee, uu, m, U, seed, Q, k, ic},
  m = Length[guess];
  ee = Rationalize[een];
  seed = Table[U[k], {k, 1, m}];
  ic = Table[{U[k], guess[[k]]}, {k, 1, m}];
  uu = FindMinimum[Total[Table[(GetBn[k, ee, seed] + k)^2, {k, 1, m}]], ic,
    WorkingPrecision → 1000, MaxIterations → 10 000, PrecisionGoal → 100, AccuracyGoal → 100];
  N[{uu[[1]], seed /. uu[[2]]}]
 ]
```

Test to see if it works. Compute the permeabilities for $\delta = 1$ for a 3 rd order BC, assuming all ones as the initial condition.

```
GetPerm[1, {1, 1, 1}]
```

$\{3.56604 \times 10^{-260}, \{1.00058, 0.872702, 4.49224\}\}$

```
pp = {0.001, 0.0025, 0.005, 0.01, 0.025, 0.05, 0.1}
```

```
{0.001, 0.0025, 0.005, 0.01, 0.025, 0.05, 0.1}
```

```
Off[FindMinimum::"precw"];
```

```
Off[FindMinimum::"sszero"];
```

```
Off[FindMinimum::"fmgz"];
```

Churn through each order, starting at $\delta = 1$ and reducing δ in small steps until $\delta = 0.001$

```
MaxOrder = 7;
iclist = {};
ttt = {};
For[nn = 1, nn ≤ MaxOrder, nn++,
 (* NewGuess=Table[1,{k,1,nn}]; *)
 NewGuess = Join[{1}, iclist];
 tt = {};
 For[k = Length[pp], k > 0, k--,
  NewGuess = GetPerm[pp[[k]], NewGuess][[2]];
  tt = Append[tt, {pp[[k]], NewGuess}];
  Print[pp[[k]], "  ", NewGuess];
  If[k == Length[pp], iclist = NewGuess];
 ];
 ttt = Append[ttt, tt];
]
```

```
For[k = 1, k ≤ MaxOrder, k++, Print[MatrixForm[ttt[[k]]]]]
```

0.1	{10.0634}
0.05	{20.0325}
0.025	{40.0165}
0.01	{100.007}
0.005	{200.003}
0.0025	{400.002}
0.001	{1000.}

0.1	{0.351313, 19.9973}
0.05	{0.188706, 39.9975}
0.025	{0.0973252, 79.9991}
0.01	{0.0395883, 200.}
0.005	{0.0198985, 400.}
0.0025	{0.00997481, 800.}
0.001	{0.00399599, 2000.}

0.1	{1.68571, 0.205825, 28.9765}
0.05	{2.99018, 0.109309, 57.3539}
0.025	{5.78115, 0.0573497, 114.392}
0.01	{14.3124, 0.0238035, 285.757}
0.005	{28.5848, 0.0120674, 571.45}
0.0025	{57.1495, 0.00607731, 1142.87}
0.001	{142.86, 0.00244168, 2857.15}

0.1	{0.801713, 2.88849, 0.163862, 37.8756}
0.05	{0.518654, 6.08207, 0.0943175, 74.2362}
0.025	{0.285096, 12.4933, 0.051874, 147.652}
0.01	{0.118297, 31.5204, 0.0221718, 368.536}
0.005	{0.0596363, 63.1283, 0.011352, 736.9}
0.0025	{0.029917, 126.301, 0.00574554, 1473.71}
0.001	{0.0119875, 315.784, 0.0023153, 3684.22}

0.1	{1.08466, 0.567794, 4.23359, 0.146273, 46.9573}
0.05	{1.46268, 0.284163, 8.73608, 0.0898108, 91.0555}
0.025	{2.50182, 0.140253, 17.6064, 0.0513426, 180.537}
0.01	{5.9424, 0.0567985, 44.1013, 0.0225322, 450.215}
0.005	{11.7947, 0.0286825, 88.227, 0.0116462, 900.108}
0.0025	{23.5444, 0.0144305, 176.466, 0.00592315, 1800.05}
0.001	{58.8295, 0.00579614, 441.175, 0.00239396, 4500.02}

0.1	{0.974642, 1.3099, 0.411777, 5.53015, 0.136558, 56.3171}
0.05	{0.817169, 2.47295, 0.206063, 11.0368, 0.0882926, 107.971}
0.025	{0.527654, 5.31367, 0.108029, 22.0233, 0.0522472, 213.325}
0.01	{0.23339, 13.826, 0.0455668, 55.0188, 0.0235108, 531.405}
0.005	{0.11886, 27.8414, 0.0232866, 110.027, 0.0122648, 1062.25}
0.0025	{0.0597818, 55.7809, 0.0117804, 220.05, 0.00626783, 2124.23}
0.001	{0.0239719, 139.52, 0.0047466, 550.112, 0.00254075, 5310.28}

0.1	{1.00707, 0.898913, 1.68988, 0.322886, 6.77713, 0.130172, 66.0146}
0.05	{1.10393, 0.561753, 3.68197, 0.174435, 13.1903, 0.087883, 125.068}
0.025	{1.54721, 0.269258, 7.79789, 0.0962378, 26.144, 0.053662, 246.115}
0.01	{3.33206, 0.103859, 19.7969, 0.0418017, 65.1895, 0.0247337, 612.333}
0.005	{6.50494, 0.0519346, 39.6752, 0.021573, 130.332, 0.0130204, 1223.81}
0.0025	{12.9299, 0.0260611, 79.3909, 0.0109676, 260.641, 0.00668574, 2447.2}
0.001	{32.2687, 0.0104599, 198.506, 0.0044323, 651.587, 0.0027181, 6117.7}

Now, we can create a function that directly gets the collection of permeabilities for any δ and ABC order :

```
GetPermAlt[dee_, enn_] := Module[{seed},
  seed = Interpolation[ttt[[enn]], InterpolationOrder → 1][dee];
  GetPerm[dee, seed][[2]]
 ]
```

Table of permeabilities needed for 0.1 R stackup case

```
MatrixForm[Table[{1 / (10 k), GetPermAlt[1 / (10 k), k]}, {k, 1, MaxOrder}]]
```

$\frac{1}{10}$	{10.0634}
$\frac{1}{20}$	{0.188706, 39.9975}
$\frac{1}{30}$	{4.37479, 0.075118, 85.8557}
$\frac{1}{40}$	{0.285096, 12.4933, 0.051874, 147.652}
$\frac{1}{50}$	{3.06077, 0.112378, 22.0265, 0.0423167, 225.43}
$\frac{1}{60}$	{0.374816, 8.17242, 0.0740027, 33.0192, 0.0371234, 319.234}
$\frac{1}{70}$	{2.40892, 0.149299, 13.8176, 0.0582172, 45.656, 0.0338673, 429.046}