Numerical matrix method for quantum periodic potentials

Felipe Le Vot, Juan J. Meléndez and Santos B. Yuste

Departamento de Física and Instituto de Computación Científica Avanzada (ICCAEX), Universidad de Extremadura, 06006 Badajoz, Spain

Mathematica version...

```
In[1]:= StringJoin[{"The running Mathematica version is ", ToString[$Version]}]

Out[1]= The running Mathematica version is
          10.1.0  for Microsoft Windows (64-bit) (March 24, 2015)
```

Kronig - Penney model

Functions that for a KP potential with lattice parameter a=1 and n_b barriers:

(I) evaluate the Hamiltonian matrix H_{nm} [Eq. (17)]

(II) solve Eq. (10) to find the eigenenergies and eigenfunctions.

The size of the system is L=n_b*a=n_b.

The positions of the nb barriers are stored in the vector (or list) xrV, and the corresponding widths and heights are stored in vectors bV and V0V, respectively.

Note that these vectors xrV, bV and V0V describe the periodic potential.

We use units such as $\hbar^2/2\mu$=1, μ being the mass of the particle.

Functions F_{nm} [Eq. (20)]

```
In[2]:= f[k_, x_, L_] := f[k, x, L] = Sin[k * Pi * x / L] / (Pi * k)
```

```
In[3]:= Fnm[n_, m_, x_, L_] := f[m - n, x, L] - f[m + n, x, L]
```

```
In[4]:= Fnn[n_, x_, L_] := x / L - f[2 * n, x, L]
```

Functions h_{nm} [Eq. (18)]

```
In[5]:= hnm[n_, m_, s_, b_, L_] := Fnm[n, m, s + b / 2, L] - Fnm[n, m, s - b / 2, L]
```

```
In[6]:= hnn[n_, s_, b_, L_] := Fnn[n, s + b / 2, L] - Fnn[n, s - b / 2, L]
```

Matrix elements H_{nm} [Eq. (17)]

```
In[7]:= HHnn[n_, L_, V0V_, xrV_, bV_] := N[(n * Pi / L) ^2 +
          Sum[V0V[[k]] * hnn[n, xrV[[k]], bV[[k]], L], {k, 1, Length[xrV]}] ]
```

```
In[8]:= HHnm[n_, m_, L_, V0V_, xrV_, bV_] :=
         Sum[V0V[[k]] * hnm[n, m, xrV[[k]], bV[[k]], L], {k, 1, Length[xrV]}] // N
```

The function *HamiltonianMatrix* provides the *N*×*N* Hamiltonian matrix for the KP potential given by xrV, bV and V0V, where *N*=Nterms

```
In[9]:= HamiltonianMatrix[Nterms_, L_, V0V_, xrV_, bV_] :=
         Table[If[n == m, HHnn[n, L, V0V, xrV, bV], HHnm[n, m, L, V0V, xrV, bV]],
          {n, 1, Nterms}, {m, 1, Nterms}]
```

The function *soluKP* provides the solution of Eq. (13).
The list of energies is provided directly by the function *EnV*. The n-th element of this matrix is the eigenvalue E_n
The matrix of coefficients $c_m^{(n)}$ of Eq. (14) are provided by *cnMatrix*. The element (n,m) of this matrix is just the coefficient $c_m^{(n)}$.

```
In[10]:= soluKP[Nterms_, L_, V0V_, xrV_, bV_] := soluKP[Nterms, L, V0V, xrV, bV] =
           Eigensystem[N[HamiltonianMatrix[Nterms, L, V0V, xrV, bV]]] // Chop //
              Transpose // Sort // Transpose
```

```
In[11]:= EnV[Nterms_, L_, V0V_, xrV_, bV_] :=
          EnV[Nterms, L, V0V, xrV, bV] = soluKP[Nterms, L, V0V, xrV, bV][[1]]
```

```
In[12]:= cnMatrix[Nterms_, L_, V0V_, xrV_, bV_] :=
          cnMatrix[Nterms, L, V0V, xrV, bV] = soluKP[Nterms, L, V0V, xrV, bV][[2]]
```

The function *psi* provides an estimate of n-th eigenfunction according the formula

$$\psi_n(x) \simeq \sum_{m=1}^{N^*} c_m^{(n)} \varphi_m(x)$$

When Nast=N^* is equal to Nterm=N, this expression is just Eq. (14).
The function ProbDensity is just $|\psi_n(x)|^2$

```
In[13]:= psi[n_, x_, Nast_, Nterms_, L_, V0V_, xrV_, bV_] := Sqrt[2 / L] *
           Sum[cnMatrix[Nterms, L, V0V, xrV, bV][[n, m]] * Sin[m * Pi * x / L], {m, 1, Nast}]
```

```
In[14]:= ProbDensity[n_, x_, Nast_, Nterms_, L_, V0V_, xrV_, bV_] :=
          Abs[psi[n, x, Nast, Nterms, L, V0V, xrV, bV]]^2
```

An example : n_b = 10, b = 1/6, V_0 =100, N=100 (case of Fig. 2 and 3)

Next, we generate the list of positions, widths and heights of the nb barriers of the KP potential

This is the case considered in Fig. 2: nb=10 barriers with width q=1/6 and height v0=100

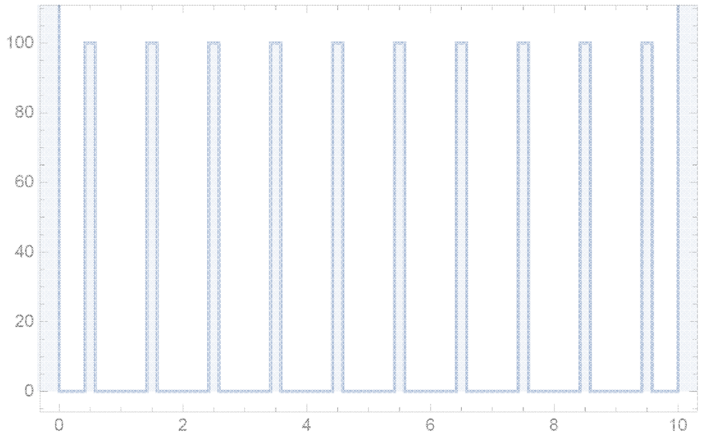
```
In[15]:= nb = 10; L = N[nb]; v0 = 100.; q = 1. / 6;
         {xrVector = Table[-1. / 2 + k, {k, 1, nb}], bVector = Table[q, {k, 1, nb}],
          V0Vector = Table[v0, {k, 1, nb}]}
```

```
Out[16]= {{0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5}, {0.166667, 0.166667, 0.166667,
           0.166667, 0.166667, 0.166667, 0.166667, 0.166667, 0.166667, 0.166667},
          {100., 100., 100., 100., 100., 100., 100., 100., 100., 100.}}
```

The next instruction generates a plot of the potential one has defined above

```
leftWall = Plot[If[x < 0, 1.2 * v0, 0],
    {x, -1, 0.01}, Filling → Bottom, PlotPoints → 100, Frame → True];
rightWall = Plot[If[x > L, 1.2 * v0, 0], {x, L - 0.01, L + 1},
   Filling → Bottom, PlotPoints → 100, Frame → True];
KPpotential = Plot[Piecewise[Table[{V0Vector[[k]],
        xrVector[[k]] - bVector[[k]] / 2 <= x <= xrVector[[k]] + bVector[[k]] / 2},
      {k, 1, nb}]], {x, 0, nb}, Filling → Bottom, PlotPoints → 2000, Frame → True];
Show[leftWall, rightWall, KPpotential, PlotRange → {{-0.1, L + 0.1}, {0, 1.05 v0}}]
```

Out[18]=



Next one chooses the value of Nterm=*N* where *N* is the number of terms employed in the expansion of the sought eigenfunction [Eq. (12)] (N=100 is the value employed in Fig. 2)

```
In[19]:= Nterms = 100;
```

The vector of the first N energies E_n is calculated by means of the function EnV

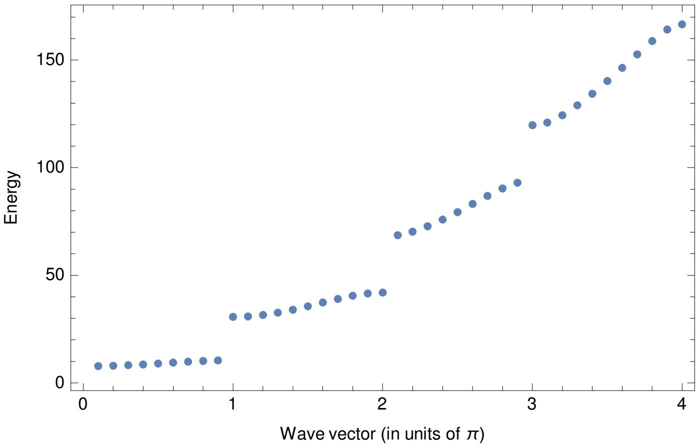
```
In[20]:= Energies = EnV[Nterms, L, V0Vector, xrVector, bVector]

Out[20]= {7.80969, 7.96994, 8.22764, 8.56813, 8.96908, 9.399, 9.81642, 10.1714, 10.4123,
          30.659, 30.8899, 31.5627, 32.6236, 33.997, 35.5927, 37.3038, 38.9949, 40.4837,
          41.5381, 41.9096, 68.6129, 70.2754, 72.7751, 75.8733, 79.3724, 83.0994,
          86.858, 90.3466, 93.0249, 119.68, 120.964, 124.329, 128.97, 134.359, 140.213,
          146.361, 152.645, 158.806, 164.156, 166.586, 189.545, 195.579, 202.89,
          210.809, 219.108, 227.683, 236.463, 245.34, 253.931, 271.389, 276.894,
          286.134, 296.098, 306.415, 316.997, 327.811, 338.84, 350.069, 361.449,
          371.172, 385.644, 397.634, 409.884, 422.35, 435.022, 447.897, 460.973,
          474.245, 487.701, 500.274, 515.356, 529.409, 543.676, 558.144, 572.81,
          587.67, 602.718, 617.938, 633.255, 652.245, 666.208, 682.103, 698.298,
          714.714, 731.336, 748.154, 765.16, 782.33, 799.56, 813.714, 835.891, 853.717,
          871.881, 890.279, 908.89, 927.706, 946.724, 965.943, 985.362, 1008.26}
```

Plot of the energies E_n obtained by means of the matrix method versus the wave number in units of π (see Fig. 2)

```
In[21]:= ListPlot[Table[{n / L, Energies[[n]]}, {n, 1, 4 * nb}], Frame → True,
          FrameLabel -> {"Wave vector (in units of π)", "Energy"}]
```

Out[21]=



Plot of the eigenfunctions

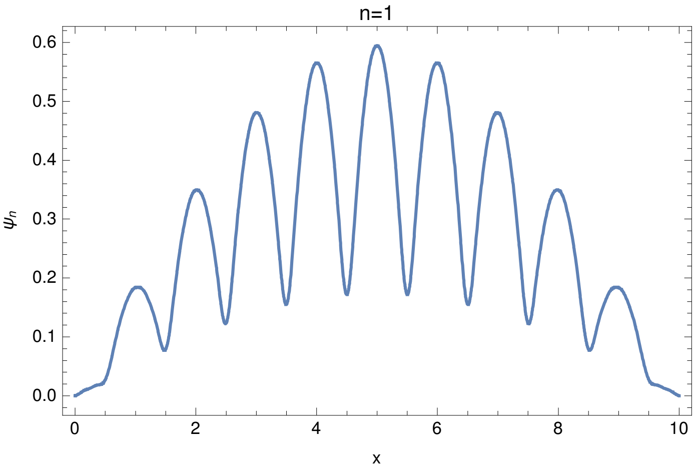
$$\psi_n(x) \simeq \sum_{m=1}^{N^*} c_m^{(n)} \varphi_m(x)$$

with Nast=N^* equal to Nterm=N .

Eigenfunction ψ_n(x) with n=nLevel=1 [see Fig. 3(a)]

```
In[22]:= nLevel = 1; Nast = Nterms;
         Plot[psi[nLevel, x, Nast, Nterms, L, V0Vector, xrVector, bVector],
          {x, 0, L}, Frame → True, FrameLabel -> {"x", "ψn"},
          PlotLabel → StringJoin[{"n=", ToString[nLevel]}]]
```

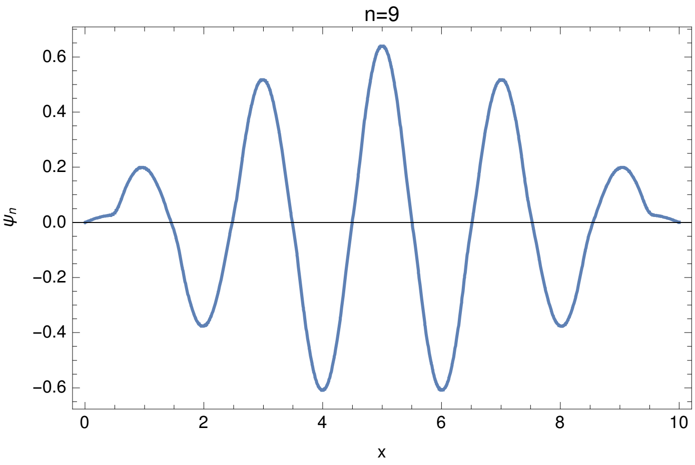
Out[22]=



Eigenfunction ψ_n(x) with n=nLevel=9 [see Fig. 3(b)]

```
In[23]:= nLevel = 9; Nast = Nterms;
         Plot[psi[nLevel, x, Nast, Nterms, L, V0Vector, xrVector, bVector],
          {x, 0, L}, Frame → True, FrameLabel -> {"x", "ψn"},
          PlotLabel → StringJoin[{"n=", ToString[nLevel]}]]
```

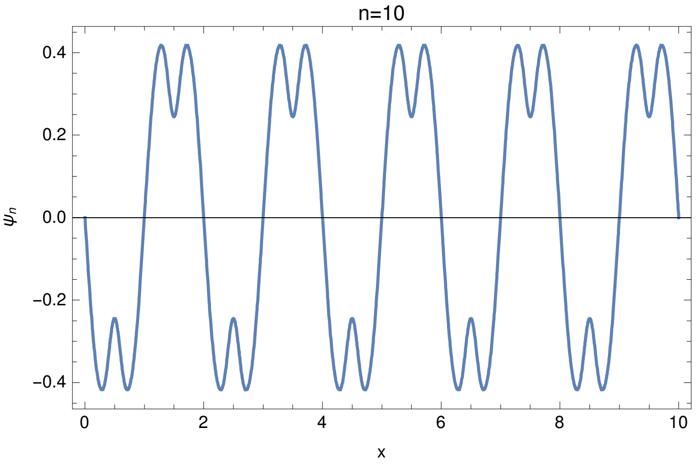
Out[23]=



Eigenfunction ψ_n(x) with n=nLevel=10 [see Fig. 3(c)]

```
In[24]:= nLevel = 10; Nast = Nterms;
         Plot[psi[nLevel, x, Nast, Nterms, L, V0Vector, xrVector, bVector],
          {x, 0, L}, Frame → True, FrameLabel -> {"x", "ψn"},
          PlotLabel → StringJoin[{"n=", ToString[nLevel]}]]
```

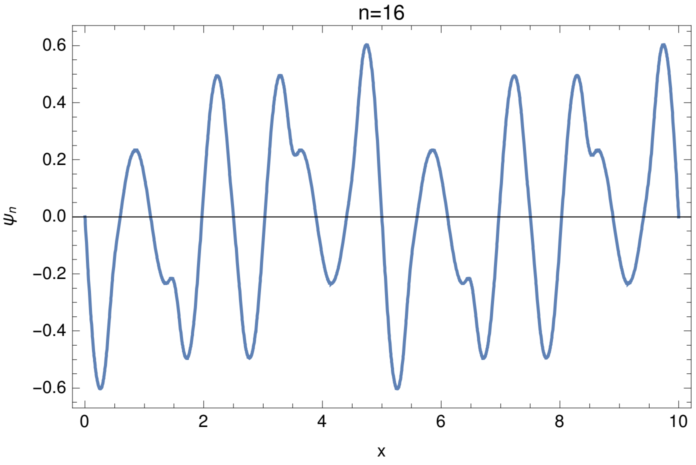
Out[24]=



Eigenfunction ψ_n(x) with n=nLevel=16 [see Fig. 3(d)]

```
In[25]:= nLevel = 16; Nast = Nterms;
         Plot[psi[nLevel, x, Nast, Nterms, L, V0Vector, xrVector, bVector],
          {x, 0, L}, Frame → True, FrameLabel -> {"x", "ψn"},
          PlotLabel → StringJoin[{"n=", ToString[nLevel]}]]
```

Out[25]=



Comparison of the energies E_n obtained with three different values of Nterm=N (N=50, 100 and 200) for n_b=10, b=1/6 and V_0=100

List of positions, widths and heights of the nb barriers of the KP potential for the case considered in Fig. 2: nb=10, b=1/6, V0=100

```
In[26]:=
nb = 10; L = N[nb]; v0 = 100.; q = 1. / 6;
{xrVector = Table[-1. / 2 + k, {k, 1, nb}], bVector = Table[q, {k, 1, nb}],
 V0Vector = Table[v0, {k, 1, nb}]}

Out[27]= {{0.5, 1.5, 2.5, 3.5, 4.5, 5.5, 6.5, 7.5, 8.5, 9.5}, {0.166667, 0.166667, 0.166667,
   0.166667, 0.166667, 0.166667, 0.166667, 0.166667, 0.166667, 0.166667},
 {100., 100., 100., 100., 100., 100., 100., 100., 100., 100.}}
```

First N energies E_n for N=50, 100, 200

```
In[28]:= Nterms = 50; energies[Nterms] = EnV[Nterms, L, V0Vector, xrVector, bVector]

Out[28]= {7.95139, 8.1051, 8.35225, 8.67844, 9.06204, 9.47297, 9.87238, 10.2144, 10.4522,
 30.9199, 31.6766, 32.3256, 33.3424, 34.6557, 36.1787, 37.8065, 39.4061,
 40.8027, 41.7816, 42.1377, 69.919, 71.4983, 73.8914, 76.8745, 80.2532,
 83.8517, 87.4713, 90.824, 93.4407, 120.68, 124.799, 127.952, 132.324, 137.454,
 143.066, 148.974, 154.996, 160.831, 165.743, 167.913, 193.216, 198.854,
 205.913, 213.68, 221.895, 230.448, 239.288, 248.384, 257.721, 273.545}

In[29]:= Nterms = 100; energies[Nterms] = EnV[Nterms, L, V0Vector, xrVector, bVector]

Out[29]= {7.80969, 7.96994, 8.22764, 8.56813, 8.96908, 9.399, 9.81642, 10.1714, 10.4123,
 30.659, 30.8899, 31.5627, 32.6236, 33.997, 35.5927, 37.3038, 38.9949, 40.4837,
 41.5381, 41.9096, 68.6129, 70.2754, 72.7751, 75.8733, 79.3724, 83.0994,
 86.858, 90.3466, 93.0249, 119.68, 120.964, 124.329, 128.97, 134.359, 140.213,
 146.361, 152.645, 158.806, 164.156, 166.586, 189.545, 195.579, 202.89,
 210.809, 219.108, 227.683, 236.463, 245.34, 253.931, 271.389, 276.894,
 286.134, 296.098, 306.415, 316.997, 327.811, 338.84, 350.069, 361.449,
 371.172, 385.644, 397.634, 409.884, 422.35, 435.022, 447.897, 460.973,
 474.245, 487.701, 500.274, 515.356, 529.409, 543.676, 558.144, 572.81,
 587.67, 602.718, 617.938, 633.255, 652.245, 666.208, 682.103, 698.298,
 714.714, 731.336, 748.154, 765.16, 782.33, 799.56, 813.714, 835.891, 853.717,
 871.881, 890.279, 908.89, 927.706, 946.724, 965.943, 985.362, 1008.26}
```

```
In[30]:= Nterms = 200; energies[Nterms] = EnV[Nterms, L, V0Vector, xrVector, bVector]

Out[30]= {7.80037, 7.96044, 8.21806, 8.55858, 8.95967, 9.38981, 9.80752, 10.1628, 10.4039,
          30.6308, 30.8618, 31.5348, 32.596, 33.9696, 35.5653, 37.2762, 38.9664, 40.4536,
          41.505, 41.8896, 68.554, 70.213, 72.7101, 75.8064, 79.3038, 83.0289, 86.7855,
          90.2715, 92.9471, 119.627, 120.909, 124.27, 128.908, 134.293, 140.144, 146.287,
          152.564, 158.712, 164.037, 166.499, 189.427, 195.437, 202.737, 210.65,
          218.943, 227.514, 236.289, 245.157, 253.726, 271.325, 276.787, 286.006,
          295.96, 306.272, 316.85, 327.661, 338.687, 349.91, 361.275, 370.948, 385.237,
          397.217, 409.466, 421.934, 434.61, 447.488, 460.568, 473.845, 487.312, 500.064,
          514.944, 529.007, 543.28, 557.752, 572.422, 587.287, 602.341, 617.571,
          632.911, 651.761, 664.982, 680.915, 697.132, 713.568, 730.206, 747.041,
          764.065, 781.258, 798.532, 812.959, 834.744, 852.612, 870.794, 889.203,
          907.819, 926.634, 945.643, 964.831, 984.135, 1007.16, 1024.09, 1043.99,
          1064.17, 1084.56, 1105.17, 1125.97, 1146.97, 1168.16, 1189.52, 1209.25,
          1233.11, 1255.07, 1277.25, 1299.64, 1322.23, 1345.02, 1368.01, 1391.19,
          1414.57, 1438.65, 1461.96, 1485.93, 1510.1, 1534.48, 1559.05, 1583.81,
          1608.78, 1633.94, 1659.29, 1685.73, 1710.63, 1736.57, 1762.72, 1789.07,
          1815.61, 1842.35, 1869.28, 1896.41, 1923.72, 1949.51, 1979.15, 2007.04,
          2035.16, 2063.47, 2091.99, 2120.7, 2149.61, 2178.7, 2207.97, 2239.69, 2267.34,
          2297.2, 2327.29, 2357.57, 2388.06, 2418.75, 2449.63, 2480.71, 2511.96,
          2541.65, 2575.36, 2607.21, 2639.28, 2671.54, 2704.01, 2736.67, 2769.53,
          2802.58, 2835.83, 2870.42, 2902.98, 2936.82, 2970.86, 3005.1, 3039.55,
          3074.18, 3109.02, 3144.05, 3179.29, 3214.54, 3250.7, 3286.52, 3322.54,
          3358.76, 3395.17, 3431.78, 3468.59, 3505.6, 3542.8, 3579.44, 3617.82, 3655.61,
          3693.6, 3731.79, 3770.18, 3808.77, 3847.55, 3886.53, 3925.71, 3966.47}
```

Table of E_n for N=50, 100, 200

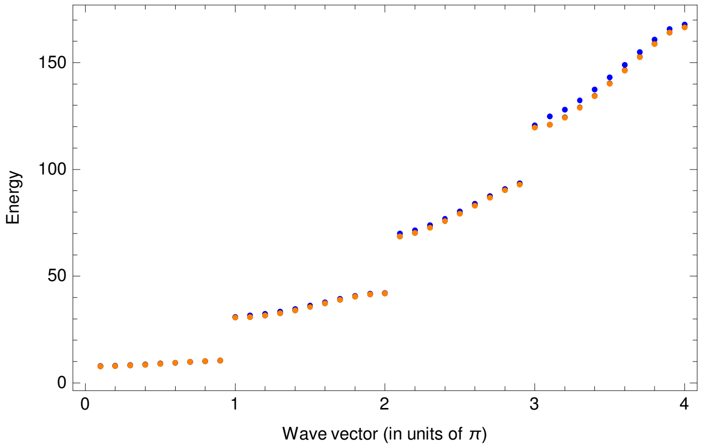
```
In[31]:= Join[{{n, "En(N=50)", "En(N=100)", "En(N=200)"}},
           Table[{n, energies[50][[n]], energies[100][[n]], energies[200][[n]]},
            {n, 1, 50}]] // TableForm
```

```
Out[31]//TableForm=
         n      En(N=50)     En(N=100)     En(N=200)
         1      7.95139      7.80969       7.80037
         2      8.1051       7.96994       7.96044
         3      8.35225      8.22764       8.21806
         4      8.67844      8.56813       8.55858
         5      9.06204      8.96908       8.95967
         6      9.47297      9.399         9.38981
         7      9.87238      9.81642       9.80752
         8      10.2144      10.1714       10.1628
         9      10.4522      10.4123       10.4039
         10     30.9199      30.659        30.6308
         11     31.6766      30.8899       30.8618
         12     32.3256      31.5627       31.5348
         13     33.3424      32.6236       32.596
         14     34.6557      33.997        33.9696
         15     36.1787      35.5927       35.5653
         16     37.8065      37.3038       37.2762
         17     39.4061      38.9949       38.9664
         18     40.8027      40.4837       40.4536
         19     41.7816      41.5381       41.505
         20     42.1377      41.9096       41.8896
         21     69.919       68.6129       68.554
         22     71.4983      70.2754       70.213
         23     73.8914      72.7751       72.7101
         24     76.8745      75.8733       75.8064
         25     80.2532      79.3724       79.3038
         26     83.8517      83.0994       83.0289
         27     87.4713      86.858        86.7855
         28     90.824       90.3466       90.2715
         29     93.4407      93.0249       92.9471
         30     120.68       119.68        119.627
         31     124.799      120.964       120.909
         32     127.952      124.329       124.27
         33     132.324      128.97        128.908
         34     137.454      134.359       134.293
         35     143.066      140.213       140.144
         36     148.974      146.361       146.287
         37     154.996      152.645       152.564
         38     160.831      158.806       158.712
         39     165.743      164.156       164.037
         40     167.913      166.586       166.499
         41     193.216      189.545       189.427
         42     198.854      195.579       195.437
         43     205.913      202.89        202.737
         44     213.68       210.809       210.65
         45     221.895      219.108       218.943
         46     230.448      227.683       227.514
         47     239.288      236.463       236.289
         48     248.384      245.34        245.157
         49     257.721      253.931       253.726
         50     273.545      271.389       271.325
```

Plot of E_n versus the wave number (in units of π) for N=50 (blue points), 100 (red points) and 200 (orange points)

In[32]:=
```
ListPlot[{
    Table[{n / L, energies[50][[n]]}, {n, 1, 4 * nb}],
    Table[{n / L, energies[100][[n]]}, {n, 1, 4 * nb}],
    Table[{n / L, energies[200][[n]]}, {n, 1, 4 * nb}]},
  PlotStyle → {Blue, Black, Orange}, Frame → True,
  FrameLabel -> {"Wave vector (in units of π)", "Energy"}]
```

Out[32]=



CPU times employed by the numerical matrix method to solve the KP model.

timeBuildSolveTotal[Nterms] is a function that provides the CPU time in seconds required to build the matrix H_{nm} and solve the eigensystem (13):

$$\sum_{m=1}^{N} H_{nm}\, c_m = E\, c_n, \quad n = 1, 2, \ldots, N$$

with *N*=Nterms. The output of this function is the list {*timeToBuildTheMatrix, timeToSolveTheEigenSystem, timeTotal*}. *timeTotal* is the total CPU required to build and solve the eigensystem (13)

In[33]:=
```
timeBuildSolveTotal[Nterms_] := {
  timeToBuildTheMatrix = Timing[Hmatrix =
      HamiltonianMatrix[Nterms, L, V0Vector, xrVector, bVector]][[1]],
  timeToSolveTheEigenSystem = Timing[
     Eigensystem[Hmatrix] / Chop // Transpose // Sort // Transpose][[1]],
  timeTotal = timeToBuildTheMatrix + timeToSolveTheEigenSystem}
```

An example: KP potential with b=1/6 and V_0=100

Case with nb=10 barriers
CPUtable[nb] saves a table with the values {*Nterms, timeToBuildTheMatrix, timeToSolveTheEigenSystem, timeTotal*}

```
In[34]:= nb = 10;
         L = N[nb]; v0 = 100.; q = 1. / 6;
         {xrVector = Table[-1. / 2 + k, {k, 1, nb}], bVector = Table[q, {k, 1, nb}],
           V0Vector = Table[v0, {k, 1, nb}]};

In[37]:= timeBuildSolveTotal[100]

Out[37]= {2.74562, 0.0156001, 2.76122}

In[38]:= CPUtable[nb] =
          Table[Flatten[{Nterms, timeBuildSolveTotal[Nterms]}], {Nterms, 50, 200, 50}];
         Insert[CPUtable[nb], {"N", "timeToBuildH", "timeToSolve", "totalCPUtime"}, 1] //
          TableForm

Out[38]//TableForm=
         N      timeToBuildH    timeToSolve    totalCPUtime
         50     0.670804        0.             0.670804
         100    2.71442         0.0156001      2.73002
         150    6.06844         0.0468003      6.11524
         200    10.7953         0.249602       11.0449
```

Case with nb=20 barriers

CPUtable[nb] saves a table with the values {*Nterms, timeToBuildTheMatrix, timeToSolveTheEigenSystem, timeTotal*}

```
In[39]:= nb = 20;
         L = N[nb]; v0 = 100.; q = 1. / 6;
         {xrVector = Table[-1. / 2 + k, {k, 1, nb}], bVector = Table[q, {k, 1, nb}],
           V0Vector = Table[v0, {k, 1, nb}]};

In[42]:= timeBuildSolveTotal[100]

Out[42]= {5.72524, 0.546004, 6.27124}

In[43]:= CPUtable[nb] =
          Table[Flatten[{Nterms, timeBuildSolveTotal[Nterms]}], {Nterms, 50, 200, 50}];
         Insert[CPUtable[nb], {"N", "timeToBuildH", "timeToSolve", "totalCPUtime"}, 1] //
          TableForm

Out[43]//TableForm=
         N      timeToBuildH    timeToSolve    totalCPUtime
         50     1.37281         0.             1.37281
         100    5.46004         0.0156001      5.47564
         150    12.6985         1.93441        14.6329
         200    22.1521         0.156001       22.3081
```

Case with nb=30 barriers

CPUtable[nb] saves a table with the values {*Nterms, timeToBuildTheMatrix, timeToSolveTheEigenSystem, timeTotal*}

```
In[44]:= nb = 30;
         L = N[nb]; v0 = 100.; q = 1. / 6;
         {xrVector = Table[-1. / 2 + k, {k, 1, nb}], bVector = Table[q, {k, 1, nb}],
           V0Vector = Table[v0, {k, 1, nb}]};
```

```
In[47]:= timeBuildSolveTotal[100]

Out[47]= {8.67366, 0.0312002, 8.70486}
```

```
In[48]:= CPUtable[nb] =
          Table[Flatten[{Nterms, timeBuildSolveTotal[Nterms]}], {Nterms, 50, 200, 50}];
         Insert[CPUtable[nb], {"N", "timeToBuildH", "timeToSolve", "totalCPUtime"}, 1] //
          TableForm

Out[48]//TableForm=
         N      timeToBuildH    timeToSolve    totalCPUtime
         50     2.01241         0.             2.01241
         100    7.92485         0.0156001      7.94045
         150    17.8309         0.0936006      17.9245
         200    32.1986         0.0468003      32.2454
```

Case with nb=40 barriers

CPUtable[nb] saves a table with the values {*Nterms, timeToBuildTheMatrix, timeToSolveTheEigenSystem, timeTotal*}

```
In[49]:= nb = 40;
         L = N[nb]; v0 = 100.; q = 1. / 6;
         {xrVector = Table[-1. / 2 + k, {k, 1, nb}], bVector = Table[q, {k, 1, nb}],
           V0Vector = Table[v0, {k, 1, nb}]};

In[52]:= timeBuildSolveTotal[100]

Out[52]= {10.9825, 0.0156001, 10.9981}
```

```
In[53]:= CPUtable[nb] =
          Table[Flatten[{Nterms, timeBuildSolveTotal[Nterms]}], {Nterms, 50, 200, 50}];
         Insert[CPUtable[nb], {"N", "timeToBuildH", "timeToSolve", "totalCPUtime"}, 1] //
          TableForm

Out[53]//TableForm=
         N      timeToBuildH    timeToSolve    totalCPUtime
         50     2.60522         0.0156001      2.62082
         100    10.3117         0.0156001      10.3273
         150    23.6966         0.0312002      23.7278
         200    42.3075         0.0624004      42.3699
```

Case with nb=50 barriers

CPUtable[nb] saves a table with the values {*Nterms, timeToBuildTheMatrix, timeToSolveTheEigenSystem, timeTotal*}

```
In[54]:= nb = 50;
         L = N[nb]; v0 = 100.; q = 1. / 6;
         {xrVector = Table[-1. / 2 + k, {k, 1, nb}], bVector = Table[q, {k, 1, nb}],
           V0Vector = Table[v0, {k, 1, nb}]};

In[57]:= timeBuildSolveTotal[100]

Out[57]= {13.9309, 0.0156001, 13.9465}
```

In[58]:=
```
CPUtable[nb] =
  Table[Flatten[{Nterms, timeBuildSolveTotal[Nterms]}], {Nterms, 50, 200, 50}];
Insert[CPUtable[nb], {"N", "timeToBuildH", "timeToSolve", "totalCPUtime"}, 1] //
  TableForm
```

Out[58]//TableForm=
```
N      timeToBuildH    timeToSolve    totalCPUtime
50     3.27602         0.0156001      3.29162
100    13.1821         0.0156001      13.1977
150    29.9678         0.0312002      29.999
200    53.1183         0.0624004      53.1807
```

We build a table with the total CPU times required to build and solve the eigensystem (13) for the values of Nterms and nb considered above. It is apparent that the CPU time scales as nb*Nterms^2

In[59]:=
```
Insert[Table[{CPUtable[10][[n, 1]], CPUtable[10][[n, 4]],
    CPUtable[20][[n, 4]], CPUtable[30][[n, 4]], CPUtable[40][[n, 4]],
    CPUtable[50][[n, 4]]}, {n, 1, Length[CPUtable[10]]}],
  {"Nterms", "nb=10", "nb=20", "nb=30", "nb=40", "nb=50"}, 1] // TableForm
```

Out[59]//TableForm=
```
Nterms   nb=10      nb=20     nb=30     nb=40     nb=50
50       0.670804   1.37281   2.01241   2.62082   3.29162
100      2.73002    5.47564   7.94045   10.3273   13.1977
150      6.11524    14.6329   17.9245   23.7278   29.999
200      11.0449    22.3081   32.2454   42.3699   53.1807
```