In[30]:=
```
(* Kronig-Penney model using transfer
  matrix as in notes by Geshjkenbein, p.6 ff *)
```

In[31]:=
```
Clear[ell, q, p, epsilon, m]
```

In[32]:=
```
ell := {{1, 1}, {I q, -I q}}   (* matrix L *)
```

In[33]:=
```
MatrixForm[ell]
```

Out[33]//MatrixForm=

$$\begin{pmatrix} 1 & 1 \\ i\,q & -i\,q \end{pmatrix}$$

In[34]:=
```
v := {{1, 0}, {epsilon, 1}}  (* matrix V,
 epsilon as defined in Section 2.2 *)
```

In[35]:=
```
MatrixForm[v]
```

Out[35]//MatrixForm=

$$\begin{pmatrix} 1 & 0 \\ \text{epsilon} & 1 \end{pmatrix}$$

In[36]:=
```
p := {{Exp[I q], 0}, {0, Exp[-I q]}}
 (* matrix P, called T in Geshjkenbein  *)
```

In[37]:=
```
MatrixForm[p]
```

Out[37]//MatrixForm=

$$\begin{pmatrix} e^{i\,q} & 0 \\ 0 & e^{-i\,q} \end{pmatrix}$$

In[38]:=
```
t := p.Inverse[ell, Method -> "CofactorExpansion"].
  v.ell  (* transfer matrix T *)
```

In[39]:=
```
MatrixForm[t]
```

Out[39]//MatrixForm=

$$\begin{pmatrix} e^{i\,q} - \frac{i\,e^{i\,q}\,\text{epsilon}}{2\,q} & -\frac{i\,e^{i\,q}\,\text{epsilon}}{2\,q} \\ \frac{i\,e^{-i\,q}\,\text{epsilon}}{2\,q} & e^{-i\,q} + \frac{i\,e^{-i\,q}\,\text{epsilon}}{2\,q} \end{pmatrix}$$

In[28]:=
```
Det[t]
```

Out[28]= 1

```
(* Eigenvectors of transfer matrix t *)
```

```
In[42]:= Eigenvectors[t]
Out[42]= {{-((epsilon + e^(2 i q) epsilon - 2 i q + 2 i e^(2 i q) q -
            i Sqrt[(-16 e^(2 i q) q^2 + (-i epsilon + i e^(2 i q) epsilon - 2
                  q - 2 e^(2 i q) q)^2)]) / (2 epsilon)), 1},
  {-((epsilon + e^(2 i q) epsilon - 2 i q + 2 i e^(2 i q) q +
            i Sqrt[(-16 e^(2 i q) q^2 + (-i epsilon + i e^(2 i q) epsilon - 2
                  q - 2 e^(2 i q) q)^2)]) / (2 epsilon)), 1}}

    (* pasted from above *)

In[43]:= va1[q_, epsilon_] :=
  {-((epsilon + e^(2 i q) epsilon - 2 i q + 2 i e^(2 i q) q -
            i Sqrt[(-16 e^(2 i q) q^2 + (-i epsilon + i e^(2 i q) epsilon - 2
                  q - 2 e^(2 i q) q)^2)]) / (2 epsilon)), 1}

In[44]:= v1[q_, epsilon_] :=
  va1[q, epsilon] / Norm[va1[q, epsilon]]   (* normalized *)

In[45]:= va2[q_, epsilon_] :=
  {-((epsilon + e^(2 i q) epsilon - 2 i q + 2 i e^(2 i q) q +
            i Sqrt[(-16 e^(2 i q) q^2 + (-i epsilon + i e^(2 i q) epsilon - 2
                  q - 2 e^(2 i q) q)^2)]) / (2 epsilon)), 1}

In[46]:= v2[q_, epsilon_] := va2[q, epsilon] / Norm[va2[q, epsilon]]


    (* Simplified form of above eigenvectors *)

In[64]:= qq[q_, epsilon_] := Cos[q] + (epsilon / (2 q)) Sin[q]

In[65]:= ww[q_, epsilon_] := -Cos[q] + (2 q / epsilon) Sin[q]

In[66]:= vv1[q_, epsilon_] := {Exp[I q] (ww[q, epsilon] +
        (2 q / epsilon) Sqrt[1 - qq[q, epsilon] ^ 2]), 1}
```

In[67]:=
```
vv2[q_, epsilon_] := {Exp[I q] (ww[q, epsilon] -
      (2 q / epsilon) Sqrt[1 - qq[q, epsilon] ^ 2]), 1}
```

In[56]:=
```
(* Check that vv1 = va1, vv2 = va2 *)
```

In[68]:=
```
va1[0.7, 0.1]
```

Out[68]= {12.5798 + 10.5958 ⅈ, 1}

In[69]:=
```
vv1[0.7, 0.1]
```

Out[69]= {12.5798 + 10.5958 ⅈ, 1}

In[70]:=
```
va2[0.7, 0.1]
```

Out[70]= {0.0465017 + 0.0391679 ⅈ, 1}

In[71]:=
```
vv2[0.7, 0.1]
```

Out[71]= {0.0465017 + 0.0391679 ⅈ, 1}

```
(* End check *)
```

In[74]:=
```
(* Modified,
normalized eigenvectors to be used in the following *)
```

In[75]:=
```
v1norm[q_, epsilon_] :=
 vv1[q, epsilon] / Norm[vv1[q, epsilon]]
```

In[76]:=
```
v2norm[q_, epsilon_] := Exp[-I q] vv2[q, epsilon] /
   Norm[vv2[q, epsilon]]  (* note factor Exp[-I q] *)
```

In[77]:=
```
(* v1norm[[1]]+v1norm[[2]] and v2norm[[1]]+
 v2norm[[2]] are complex conjugate,
   but only with factor Exp[-I q] in v2norm
 included. This is the modification. *)
```

In[79]:=
```
v1norm[0.7, 0.1][[1]] + v1norm[0.7, 0.1][[2]]
```

Out[79]= 0.82412 + 0.64303 ⅈ

In[80]:=
```
v2norm[0.7, 0.1][[1]] + v2norm[0.7, 0.1][[2]]
```

Out[80]= $0.82412 - 0.64303\,i$

In[81]:=
```
(* alternative form of v2norm, further simplified *)
```

In[82]:=
```
vv2alt[q_, epsilon_] := {ww[q, epsilon] -
   (2 q / epsilon) Sqrt[1 - qq[q, epsilon] ^ 2], Exp[-I q]}
```

In[83]:=
```
v2normalt[q_, epsilon_] :=
 vv2alt[q, epsilon] / Norm[vv2alt[q, epsilon]]
```

In[84]:=
```
v2normalt[0.7, 0.1][[1]] + v2normalt[0.7, 0.1][[2]]
```

Out[84]= $0.82412 - 0.64303\,i$

```
(* Eigenvalues mu of transfer matrix t,
cp. Geshkenbein, Eq. (50) *)
```

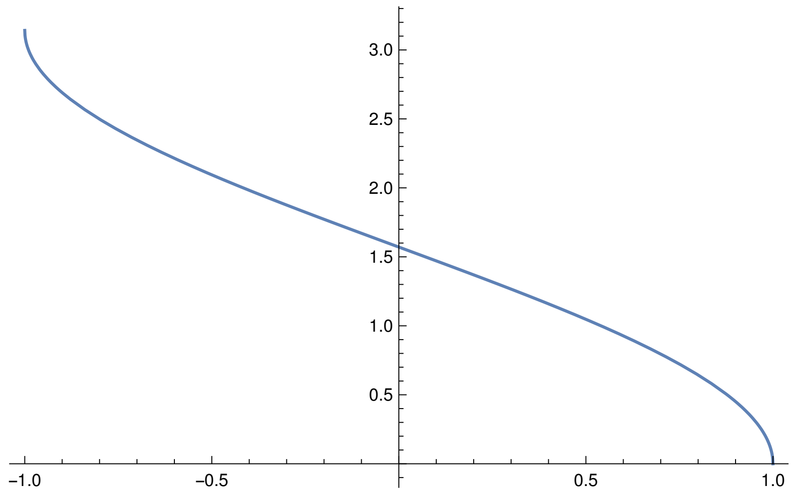
In[88]:=
```
Eigenvalues[t]
```

Out[88]=

$$\left\{\frac{1}{4\,q} e^{-i\,q}\left(i\,\text{epsilon} - i\,e^{2\,i\,q}\,\text{epsilon} + 2\,q + 2\,e^{2\,i\,q}\,q - \sqrt{\left(-16\,e^{2\,i\,q}\,q^2 + \left(-i\,\text{epsilon} + i\,e^{2\,i\,q}\,\text{epsilon} - 2\,q - 2\,e^{2\,i\,q}\,q\right)^2\right)}\right),\ \frac{1}{4\,q} e^{-i\,q}\left(i\,\text{epsilon} - i\,e^{2\,i\,q}\,\text{epsilon} + 2\,q + 2\,e^{2\,i\,q}\,q + \sqrt{\left(-16\,e^{2\,i\,q}\,q^2 + \left(-i\,\text{epsilon} + i\,e^{2\,i\,q}\,\text{epsilon} - 2\,q - 2\,e^{2\,i\,q}\,q\right)^2\right)}\right)\right\}$$

```
(* => mu = exp(+/- i k),
cos(k) = cos(q) + epsilon/(2q) sin(q) =: Q(q) *)
```

In[90]:=
```
Plot[ArcCos[x], {x, -1, 1}]
```

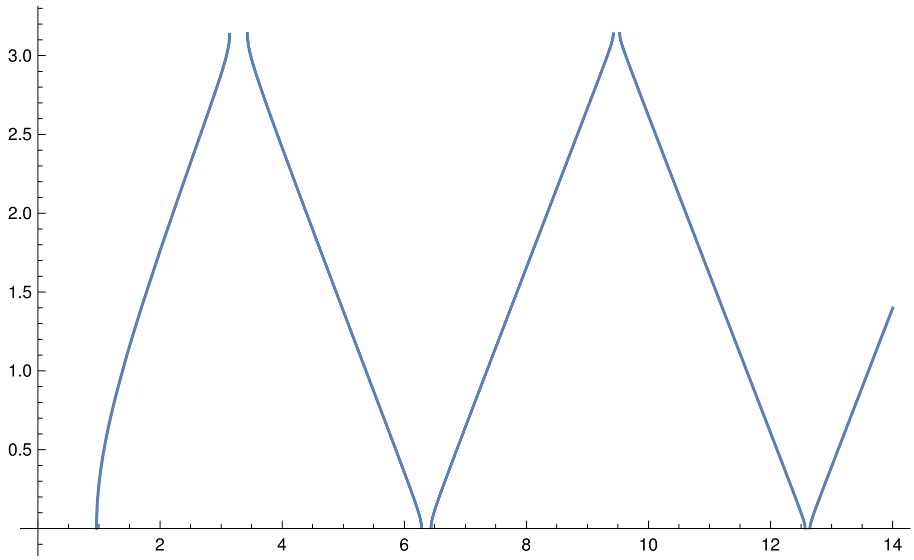
Out[90]=


In[91]:=
```
kk[q_, epsilon_] :=
 ArcCos[qq[q, epsilon]]              (* 0 < k < Pi *)
```

```
(* Illustration: plot band structure q(k) =
 omega(k)/v for -Pi<k<Pi in reduced zone scheme *)
(* for epsilon = 1 *)
```

In[93]:=
```
Plot[kk[q, 1], {q, 0, 14}]     (* function k(q),
range 0 < k < Pi, epsilon = 1 *)
```

Out[93]=

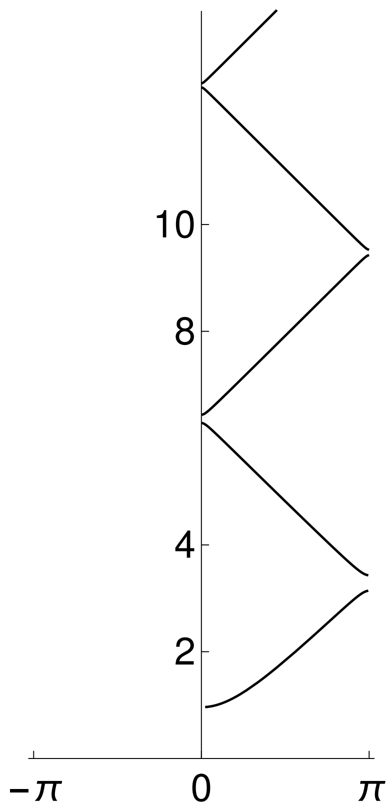


In[94]:=
```
kq[q_, epsilon_] :=
 If[Abs[qq[q, epsilon]] < 1, kk[q, epsilon], 100]
 (* k(q), allow only |q|<1 *)
```

```
(* use parametric plot to plot inverse function q(k) *)
```

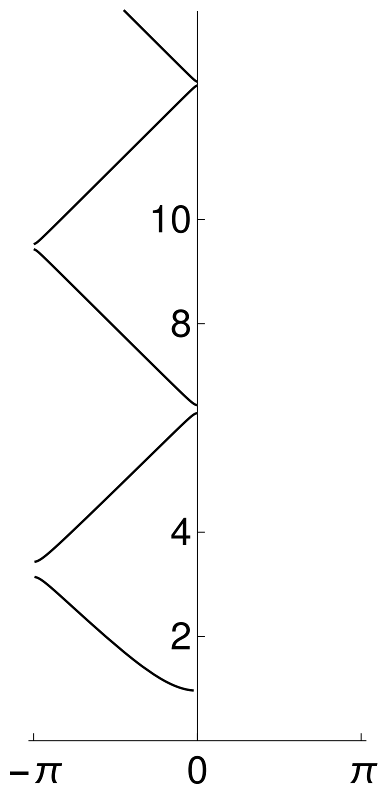
In[100]:=
```
pos = ParametricPlot[{kq[q, 1], q}, {q, 0, 14},
  PlotRange → {{-Pi - 0.1, Pi + 0.1}, {0, 14}},
  PlotStyle → {{Black, Thickness[0.007]}},
  AxesStyle → Directive[20],
  Ticks → {{-Pi, 0, Pi}, {2, 4, 8, 10}}]
```

Out[100]=



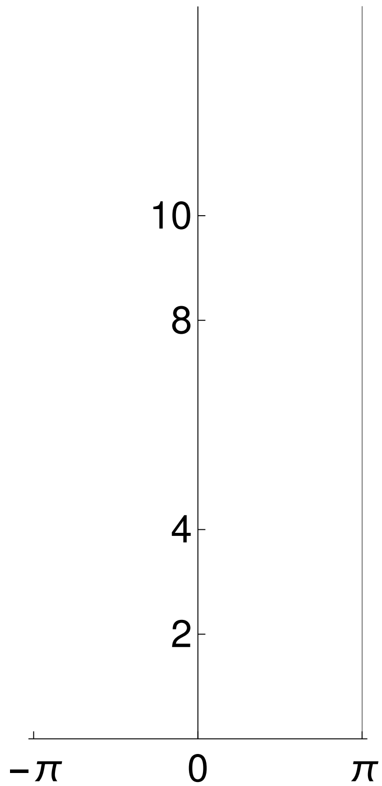
In[101]:=
```
neg = ParametricPlot[{-kq[q, 1], q}, {q, 0, 14},
  PlotRange → {{-Pi - 0.1, Pi + 0.1}, {0, 14}},
  PlotStyle → {{Black, Thickness[0.007]}},
  AxesStyle → Directive[20],
  Ticks → {{-Pi, 0, Pi}, {2, 4, 8, 10}}]
```

Out[101]=



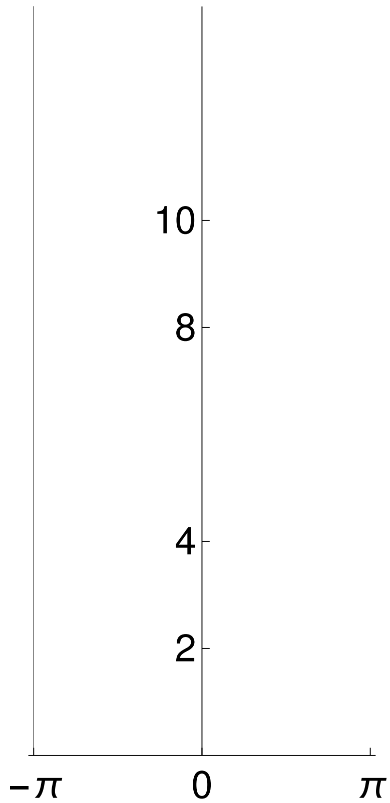
In[102]:=
```
pipos = ParametricPlot[{Pi, q}, {q, 0, 14},
  PlotRange → {{-Pi - 0.1, Pi + 0.1}, {0, 14}},
  PlotStyle → {{Black, Thickness[0.0015]}},
  AxesStyle → Directive[20],
  Ticks → {{-Pi, 0, Pi}, {2, 4, 8, 10}}]
```

Out[102]=



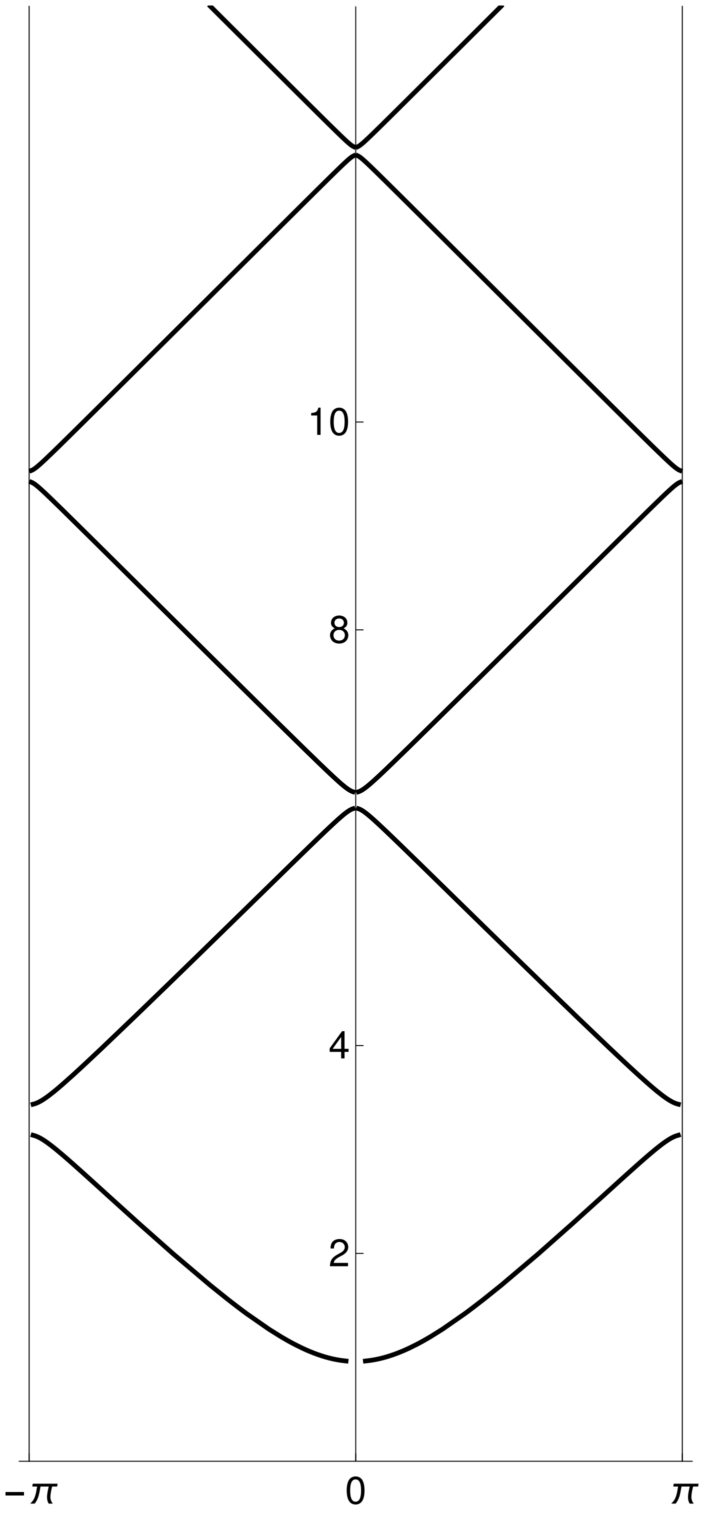
In[103]:=
```
pineg = ParametricPlot[{-Pi, q}, {q, 0, 14},
   PlotRange → {{-Pi - 0.1, Pi + 0.1}, {0, 14}},
   PlotStyle → {{Black, Thickness[0.0015]}},
   AxesStyle → Directive[20],
   Ticks → {{-Pi, 0, Pi}, {2, 4, 8, 10}}]
```

Out[103]=



In[104]:=
```
band = Show[neg, pos, pipos, pineg]
```

Out[104]=



```
In[105]:= Export["bs.png", band, ImageResolution → 300]

Out[105]= bs.png

(* End Illustration *)



(* Bloch functions Phi(x)
 with additional normalization *)

(* allow only |q|<1 *)

(* in what follows: index 1 = +k, index 2 = -k *)

In[106]:= exp1[q_, epsilon_, n_] := If[Abs[qq[q, epsilon]] < 1,
            Exp[I kk[q, epsilon] n], 0]   (* k1 > 0 *)

In[107]:= exp2[q_, epsilon_, n_] := If[Abs[qq[q, epsilon]] < 1,
            Exp[-I kk[q, epsilon] n], 0]  (* k2 = -k1 < 0 *)

(* phi1raw, phi2raw not correctly normalized,
missing normalization factors norm1, norm2 below *)

In[108]:= phi1raw[x_, n_, q_, epsilon_] :=
           exp1[q, epsilon, n] (v1norm[q, epsilon][[1]] Exp[I q (x - n)] +
                v1norm[q, epsilon][[2]] Exp[-I q (x - n)]) /
             (v1norm[q, epsilon][[1]] + v1norm[q, epsilon][[2]])

In[109]:= phi2raw[x_, n_, q_, epsilon_] :=
           exp2[q, epsilon, n] (v2norm[q, epsilon][[1]] Exp[I q (x - n)] +
                v2norm[q, epsilon][[2]] Exp[-I q (x - n)]) /
             (v2norm[q, epsilon][[1]] + v2norm[q, epsilon][[2]])



(* functions u(x) for 0<x<1, then periodically extended *)
```

In[110]:=
```
u1raw[x_, q_, epsilon_] :=
 (v1norm[q, epsilon][[1]] Exp[I (q - kk[q, epsilon]) (x - 1)] +
    v1norm[q, epsilon][[2]]
     Exp[-I (q + kk[q, epsilon]) (x - 1)]) /
  (v1norm[q, epsilon][[1]] + v1norm[q, epsilon][[2]])
```

In[111]:=
```
norm1[q_, epsilon_] := Sqrt[NIntegrate[u1raw[x, q, epsilon]
    Conjugate[u1raw[x, q, epsilon]], {x, 0, 1}]]
```

In[112]:=
```
u1[x_, q_, epsilon_] :=
 u1raw[x, q, epsilon] / norm1[q, epsilon]
```

In[113]:=
```
NIntegrate[u1[x, 0.3, 0.7] Conjugate[u1[x, 0.3, 0.7]],
 {x, 0, 1}]   (* u1 correctly normalized *)
```

Out[113]= 1.

In[114]:=
```
u2raw[x_, q_, epsilon_] :=
 (v2norm[q, epsilon][[1]] Exp[I (q + kk[q, epsilon]) (x - 1)] +
    v2norm[q, epsilon][[2]]
     Exp[-I (q - kk[q, epsilon]) (x - 1)]) /
  (v2norm[q, epsilon][[1]] + v2norm[q, epsilon][[2]])
```

In[115]:=
```
norm2[q_, epsilon_] := Sqrt[NIntegrate[u2raw[x, q, epsilon]
    Conjugate[u2raw[x, q, epsilon]], {x, 0, 1}]]
```

In[116]:=
```
u2[x_, q_, epsilon_] :=
 u2raw[x, q, epsilon] / norm2[q, epsilon]
```

In[117]:=
```
NIntegrate[u2[x, 0.3, 0.7] Conjugate[u2[x, 0.3, 0.7]],
 {x, 0, 1}]   (* u2 correctly normalized *)
```

Out[117]= 1.

```
(* functions c(q), d(q) *)
```

In[121]:=
```
u1prime[x_, q_, epsilon_] :=
 Derivative[1, 0, 0][u1raw][x, q, epsilon] / norm1[q, epsilon]
```

In[119]:=
```
u2prime[x_, q_, epsilon_] :=
 Derivative[1, 0, 0][u2raw][x, q, epsilon] / norm2[q, epsilon]

c[q_, epsilon_] := u1[0, q, epsilon]  (* cal C in thesis *)

(* c real and equal for (1), (2) *)
```

In[122]:= `c[0.9, 0.2]`

Out[122]= $0.982811 - 5.4557 \times 10^{-17}\, i$

In[123]:= `u2[0, 0.9, 0.2]`

Out[123]= $0.982811 - 1.85494 \times 10^{-15}\, i$

In[124]:= `c[2, 3]`

Out[124]= $0.762883 + 3.46945 \times 10^{-17}\, i$

In[125]:= `u2[0, 2, 3]`

Out[125]= $0.762883 - 4.16334 \times 10^{-17}\, i$

In[126]:=
```
d[q_, epsilon_] :=
 u1prime[0, q, epsilon]   (* cal D in thesis *)

(* d complex and conjugate for (1), (2) *)
```

In[127]:= `d[0.9, 0.2]`

Out[127]= $0.0982811 + 0.0269644\, i$

In[128]:= `u2prime[0, 0.9, 0.2]`

Out[128]= $0.0982811 - 0.0269644\, i$

In[129]:= `d[2, 3]`

Out[129]= $1.14432 + 0.624518\, i$

In[130]:= `u2prime[0, 2, 3]`

Out[130]= $1.14432 - 0.624518\, i$

```
(* Note: For our calculation we
  need C and D as functions of k instead *)
(* of q=omega. For this we need the
  function q(k) which is the inverse  *)
(* of the function k(q). See illustration
 at the beginnng of the notebook. *)
(* If we have q(k) then c[k, epsilon] =
 u1[0, q(k), epsilon] and *)
(* d[k, epsilon] = uprime1[0, q(k), epsilon] *)


(* Illustrations and side calculations *)


(* Properties of u1, u2 and derivatives *)
(* Show u1(0)=u1(1)=u2(0)=u2(1) and real *)
```

In[161]:= `Abs[qq[0.7, 0.3]]  (* needs to be <1 *)`

Out[161]= 0.902889

In[162]:= `u1[0, 0.7, 0.3] // N`

Out[162]= $0.975105 - 1.62388 \times 10^{-16}\, i$

In[163]:= `u1[1, 0.7, 0.3] // N`

Out[163]= $0.975105 + 0.\, i$

In[170]:= `u2[0, 0.7, 0.3] // N`

Out[170]= $0.975105 - 2.16517 \times 10^{-16}\, i$

In[171]:= `u2[1, 0.7, 0.3] // N`

Out[171]= $0.975105 + 5.41292 \times 10^{-17}\, i$

In[172]:= `Abs[qq[0.9, 0.2]]  (* needs to be <1 *)`

Out[172]= 0.708646

In[173]:=
```
u1[0, 0.9, 0.2] // N
```
Out[173]= $0.982811 - 5.4557 \times 10^{-17}\, i$

In[174]:=
```
u1[1, 0.9, 0.2] // N
```
Out[174]= $0.982811 + 0.\, i$

In[175]:=
```
u2[0, 0.9, 0.2] // N
```
Out[175]= $0.982811 - 1.85494 \times 10^{-15}\, i$

In[176]:=
```
u2[1, 0.9, 0.2] // N
```
Out[176]= $0.982811 + 0.\, i$

In[177]:=
```
Abs[qq[0.3, 0]]  (* needs to be <1 *)
```
Out[177]= 0.955336

In[142]:=
```
(* Show u1=u2=1 for epsilon=0 *)
```
In[143]:=
```
u1[0.9, 0.3, 0.0000000001]
```
Out[143]= $1. - 3.62765 \times 10^{-13}\, i$

In[144]:=
```
u2[0.9, 0.3, 0.0000000001]
```
Out[144]= $1. + 4.1371 \times 10^{-8}\, i$

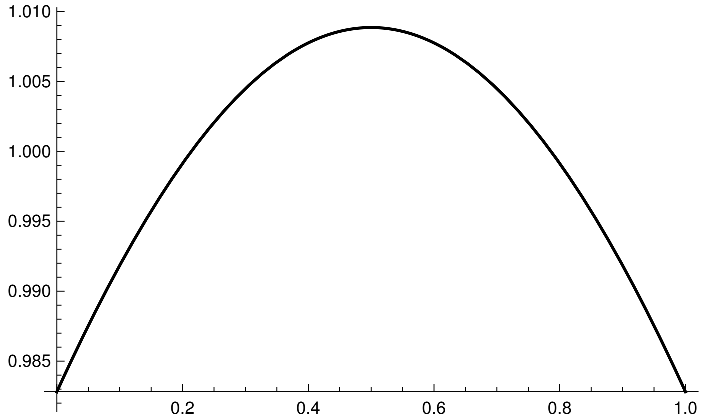
In[145]:=
```
(* Show: u2 = Conjugate[u1] if Abs[qq]<1 *)
```
In[178]:=
```
Abs[qq[0.9, 0.2]]  (* needs to be <1 *)
```
Out[178]= 0.708646

In[179]:=
```
u1[0.3, 0.9, 0.2]
```
Out[179]= $1.00448 + 0.00233511\, i$

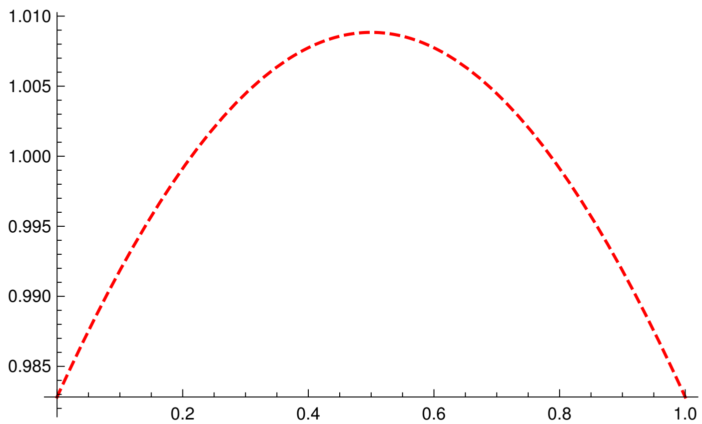
In[180]:=
```
u2[0.3, 0.9, 0.2]
```
Out[180]= $1.00448 - 0.00233511\, i$

In[151]:=
```
(* plot real part of u(x) for epsilon = 0.2 *)
```

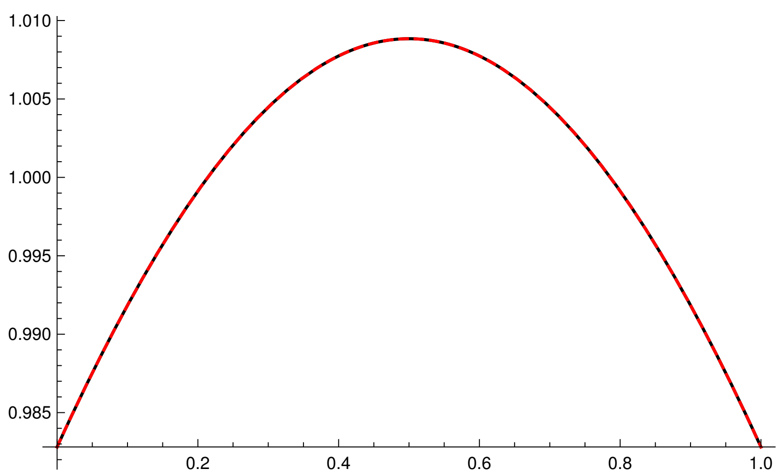
In[181]:=
```
pu1 = Plot[Re[u1[x, 0.9, 0.2]], {x, 0, 1}, PlotStyle → Black]
```

Out[181]=


In[182]:=
```
pu2 = Plot[Re[u2[x, 0.9, 0.2]],
  {x, 0, 1}, PlotStyle → {Red, Dashed}]
```

Out[182]=


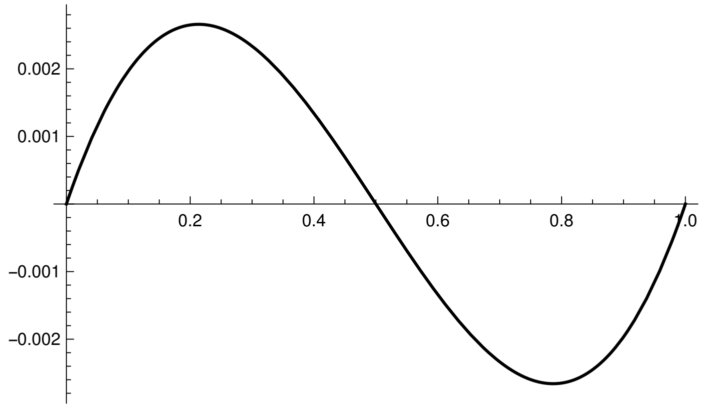
In[183]:=
```
Show[pu1, pu2]
```

Out[183]=



```
(* plot imaginary part of u(x) *)
```

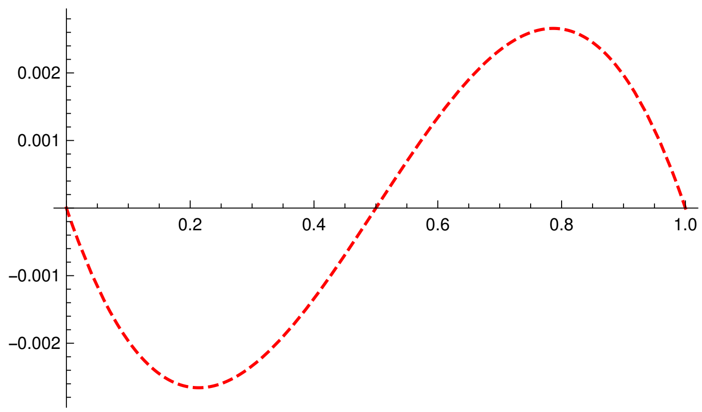
In[184]:=
```
pu3 = Plot[Im[u1[x, 0.9, 0.2]], {x, 0, 1}, PlotStyle → Black]
```

Out[184]=



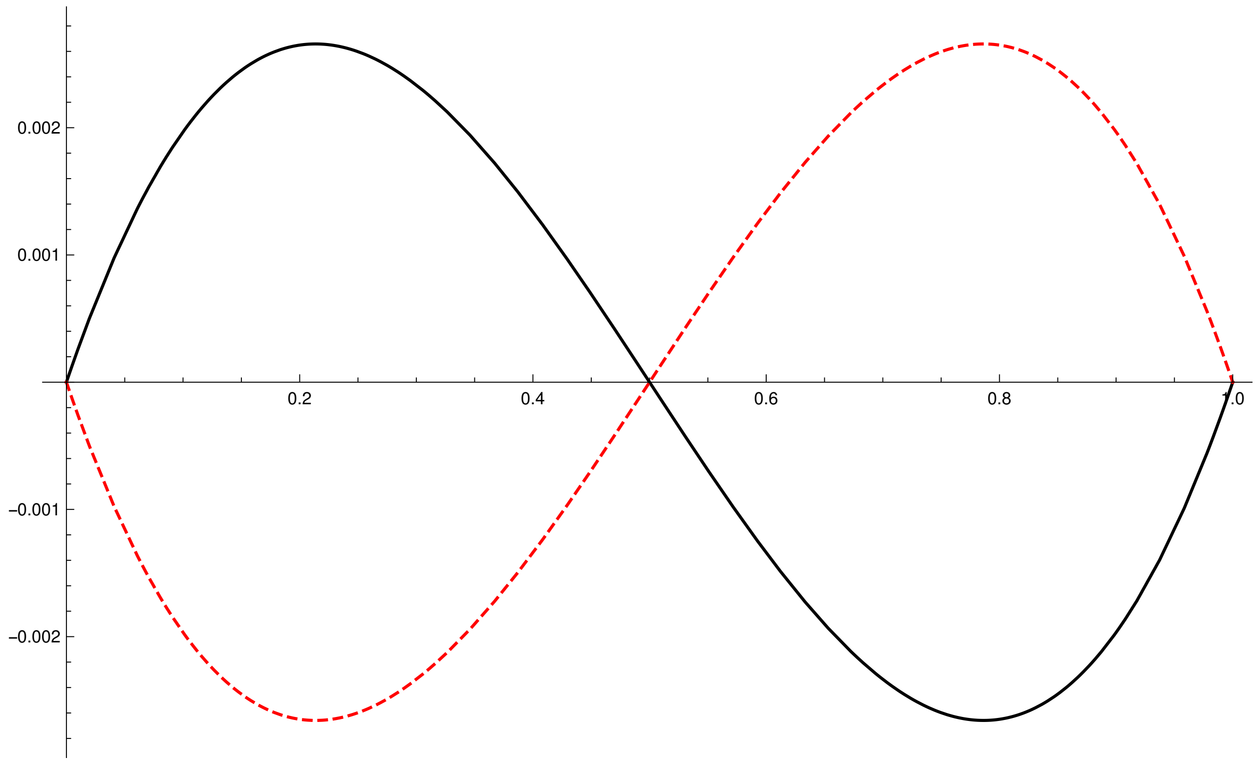
In[185]:=
```
pu4 = Plot[Im[u2[x, 0.9, 0.2]],
  {x, 0, 1}, PlotStyle → {Red, Dashed}]
```

Out[185]=



In[186]:=
```
Show[pu3, pu4]
```

Out[186]=




```
(* Illustration: plot u(x) for epsilon = 1 and q=2 *)
```

In[193]:=
```
Abs[qq[2, 1]] // N (* needs to be <1 *)
```

Out[193]= 0.188822

In[194]:=
```
u1[0.4, 2, 1]
```

Out[194]= 1.05025 + 0.0207882 i

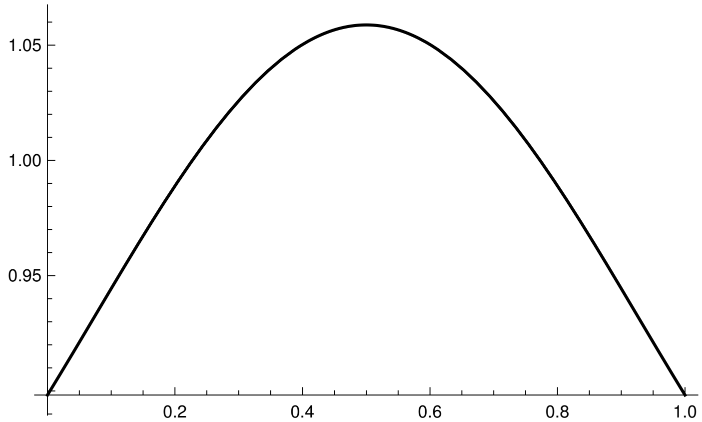
In[195]:=
```
u2[0.4, 2, 1]
```

Out[195]= 1.05025 - 0.0207882 i

```
(* plot real part of u(x) *)
```

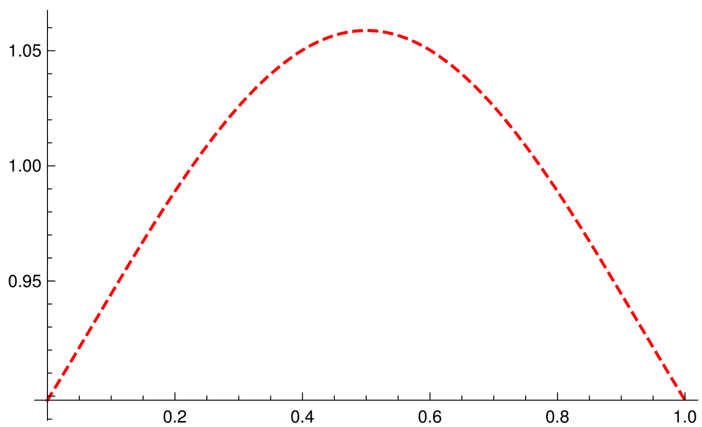
In[196]:=
```
pu5 = Plot[Re[u1[x, 2, 1]], {x, 0, 1}, PlotStyle → Black]
```

Out[196]=



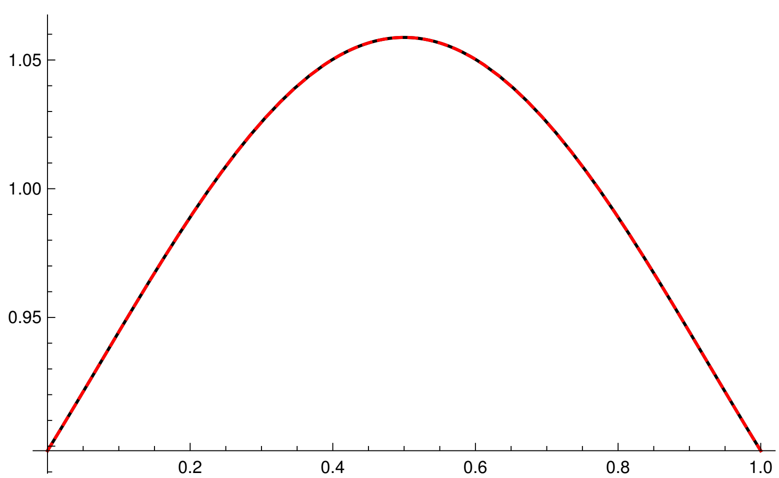
In[197]:=
```
pu6 = Plot[Re[u2[x, 2, 1]],
  {x, 0, 1}, PlotStyle → {Red, Dashed}]
```

Out[197]=



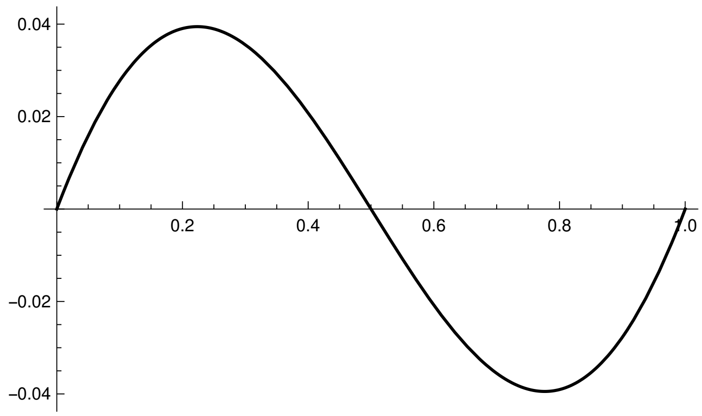
In[198]:=
```
Show[pu5, pu6]
```

Out[198]=

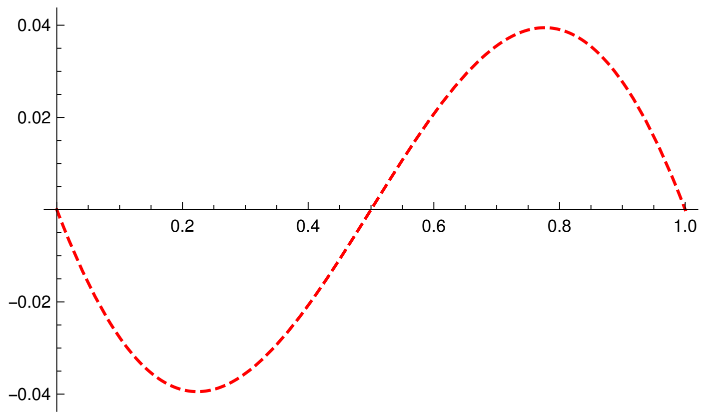



```
(* plot imaginary part of u(x) *)
```

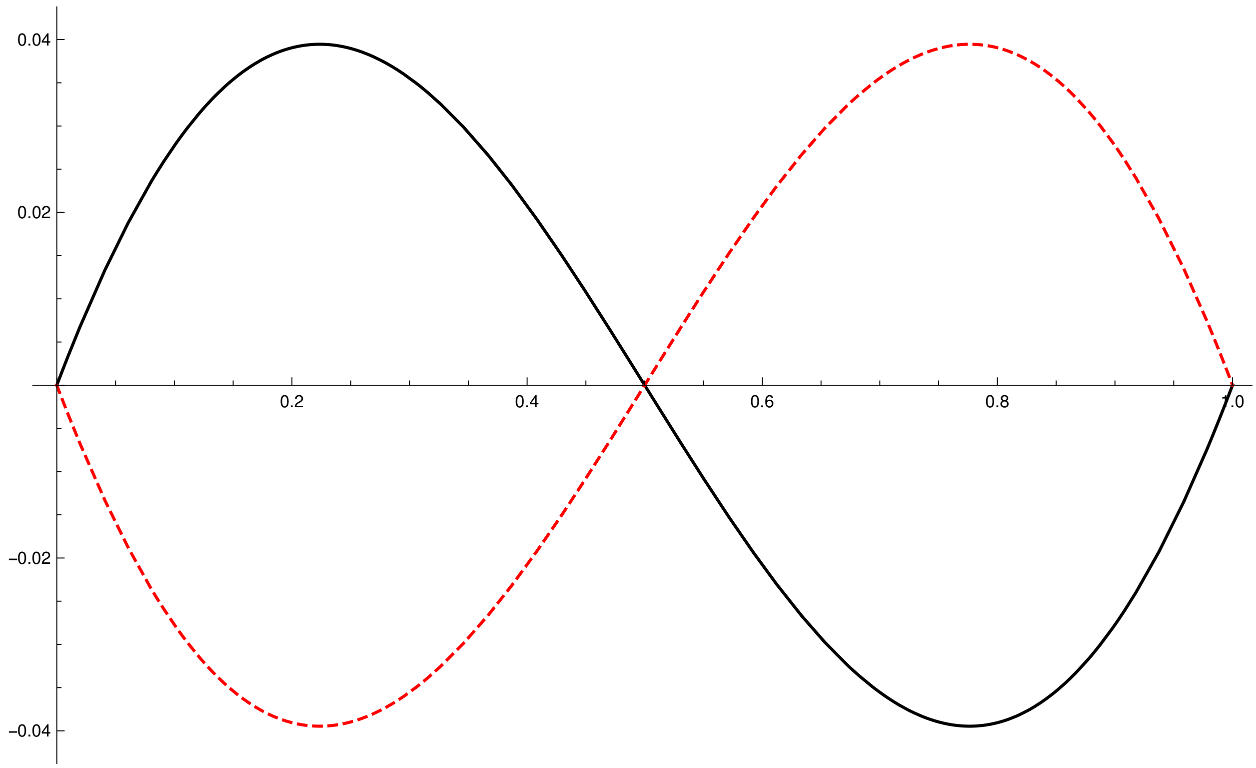
In[199]:=
```
pu7 = Plot[Im[u1[x, 2, 1]], {x, 0, 1}, PlotStyle → Black]
```

Out[199]=


In[200]:=
```
pu8 = Plot[Im[u2[x, 2, 1]],
  {x, 0, 1}, PlotStyle → {Red, Dashed}]
```

Out[200]=


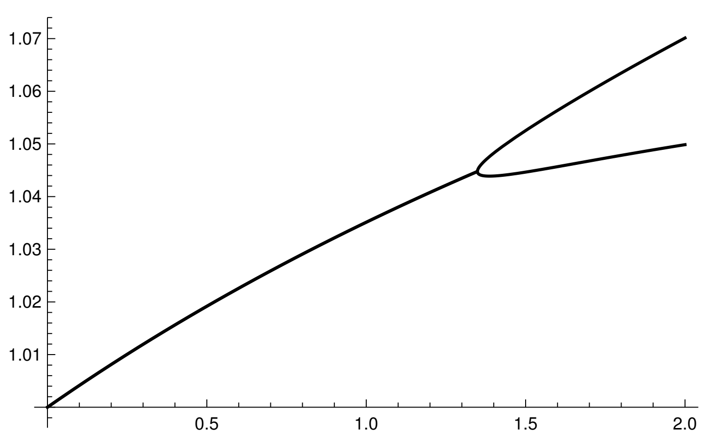
In[201]:=
```
Show[pu7, pu8]
```

Out[201]= 

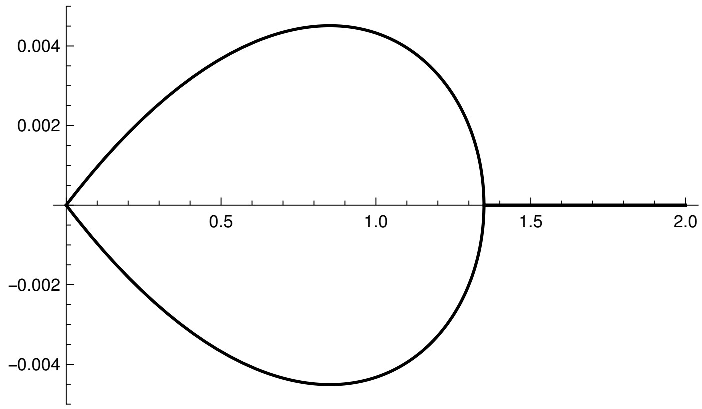

```
Abs[qq[1.1, 1.348]]
(* epsilon=1.348 is critical value for q=1.1 *)
```

0.999663

```
Plot[{Re[u1[0.4, 1.1, eps]], Re[u2[0.4, 1.1, eps]]},
 {eps, 0, 2}, PlotStyle → Black]
```

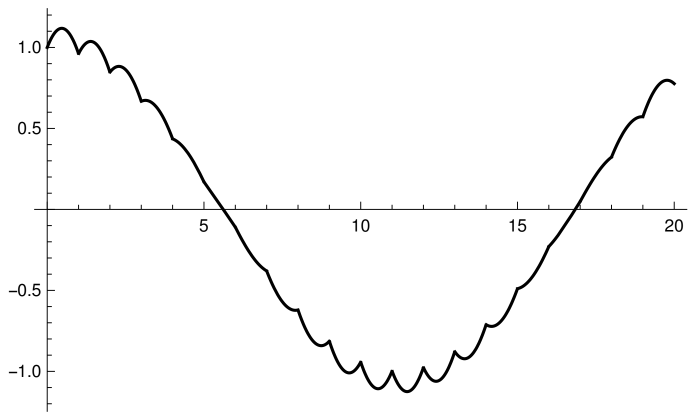


```
Plot[{Im[u1[0.4, 1.1, eps]], Im[u2[0.4, 1.1, eps]]},
 {eps, 0, 2}, PlotStyle → Black]
```

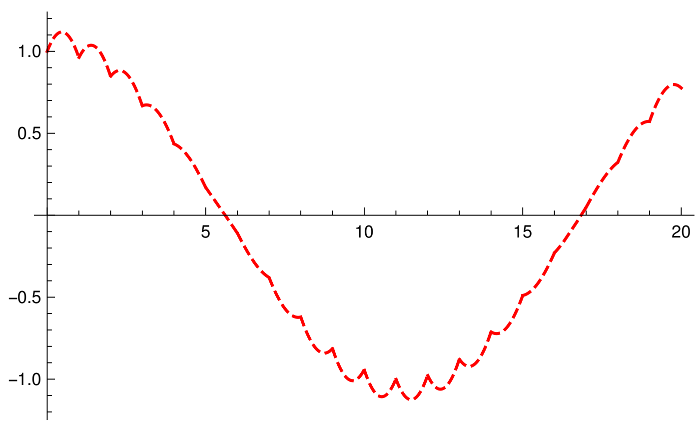



```
(* Show: u2'[0] = Conjugate[u1'[0]] if Abs[qq]<1 *)
Abs[qq[0.9, 0.2]]  (* needs to be <1 *)
0.708646

u1prime[0, 0.9, 0.2]
0.0982811 + 0.0269644 i

u2prime[0, 0.9, 0.2]
0.0982811 - 0.0269644 i

Abs[qq[2, 3]] // N  (* needs to be <1 *)
0.265826

u1prime[0, 2, 3]
1.14432 + 0.624518 i

u2prime[0, 2, 3]
1.14432 - 0.624518 i

(* END properties of u1, u2 and derivatives *)
```

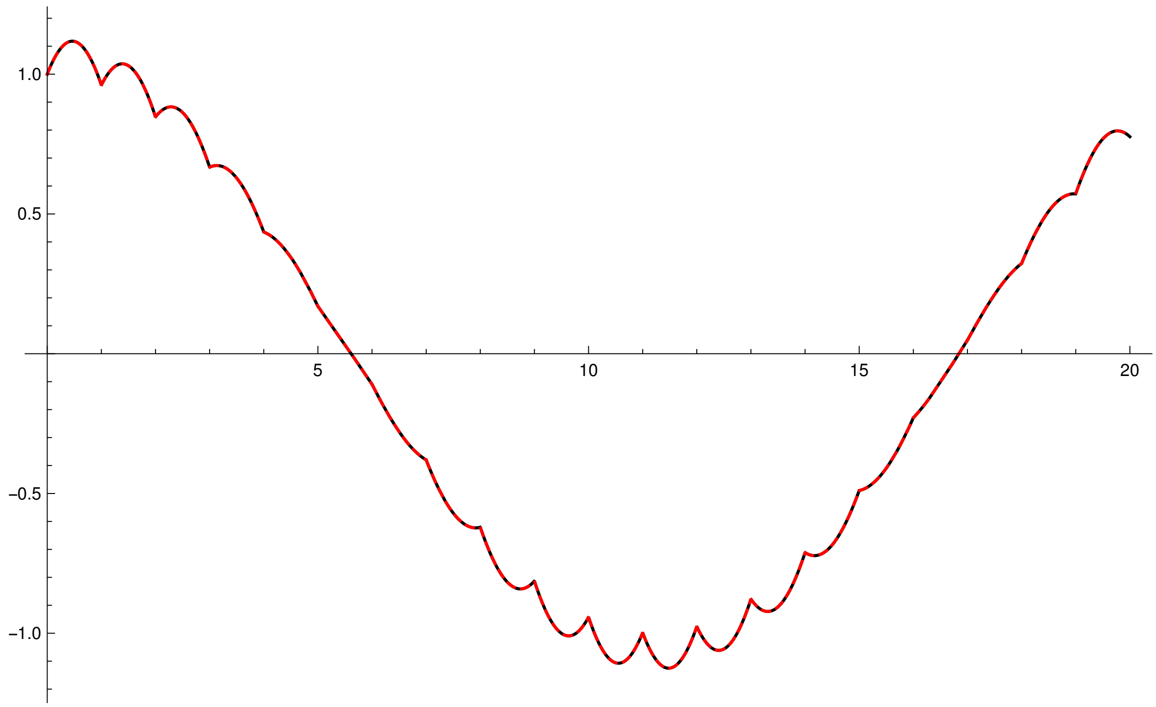
```
(* Plot real and imaginary
 parts of Bloch functions Phi_raw(x) *)
(* Show Phi_2_raw = complex conjugate Phi_1_raw *)
Abs[qq[1, 1]] // N (* needs to be <1 *)
0.961038
p1 = Plot[Re[phi1raw[x, Ceiling[x], 1, 1]],
  {x, 0, 20}, PlotStyle → Black]
```



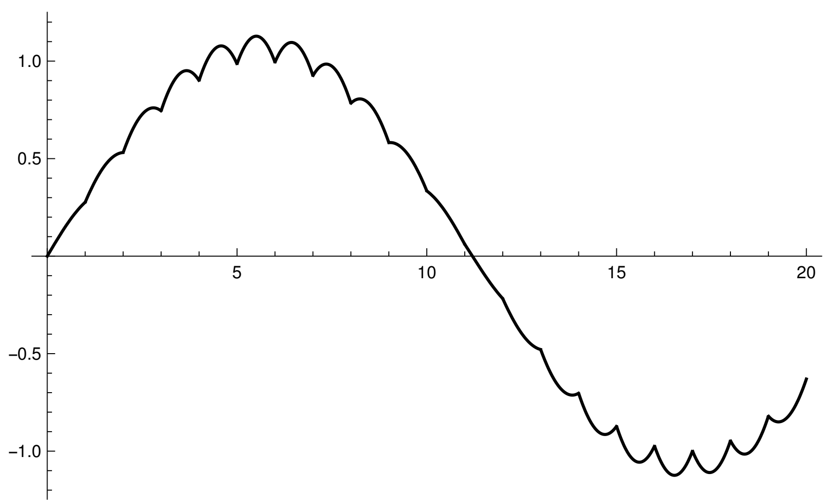

```
p2 = Plot[Re[phi2raw[x, Ceiling[x], 1, 1]], {x, 0, 20},
  PlotStyle → {Red, Dashed}] (* same as Re(phi1raw) *)
```



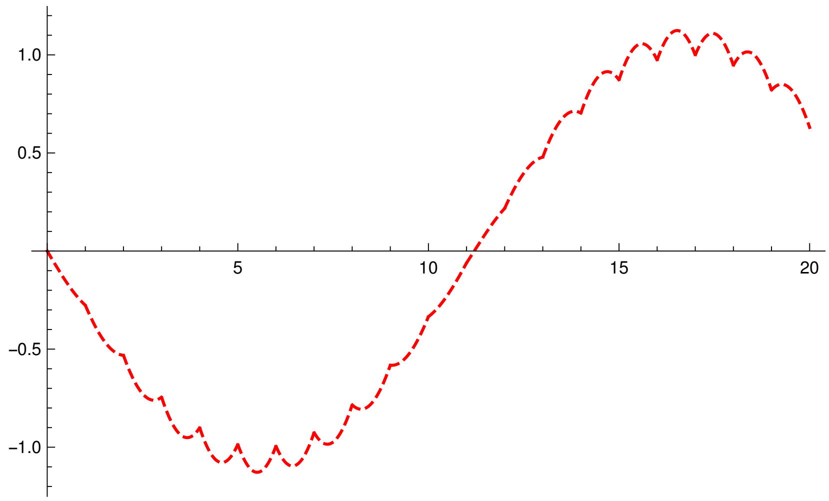
```
Show[p1, p2]
```



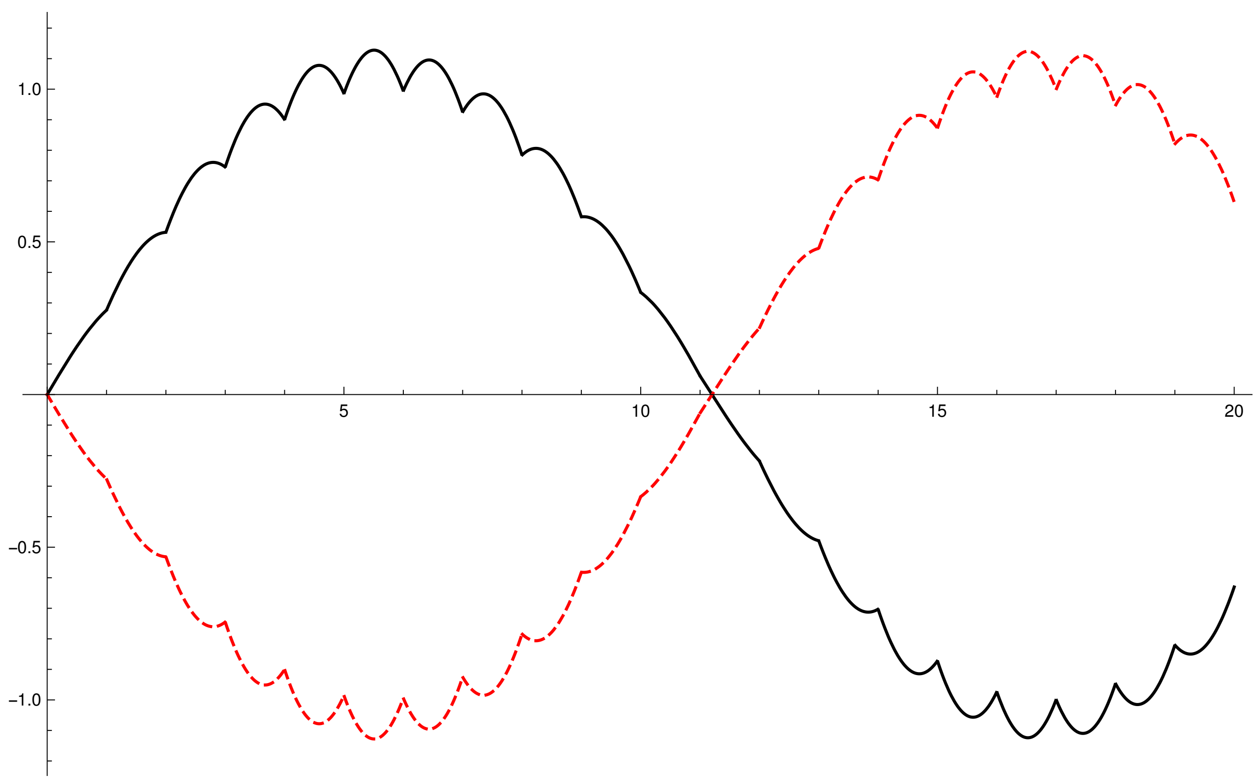

```
p3 = Plot[Im[phi1raw[x, Ceiling[x], 1, 1]],
  {x, 0, 20}, PlotStyle → Black]
```



```
p4 = Plot[Im[phi2raw[x, Ceiling[x], 1, 1]], {x, 0, 20},
  PlotStyle → {Red, Dashed}] (* -Im(phi1raw) *)
```




```
Show[p3, p4]
```



```
(* Show that phi1raw(x=n,n,q,epsilon) = Exp(i k n) *)
phi1raw[7, 7, 0.9, 0.2] // N
0.696237 - 0.717812 i
Exp[I kk[0.9, 0.2] 7]  // N
0.696237 - 0.717812 i
phi1raw[8, 8, 2, 3] // N
-0.549437 - 0.835535 i
Exp[I kk[2, 3] 8]  // N
-0.549437 - 0.835535 i

(* Show that phi2raw(x=n,n,q,epsilon) = Exp(-i k n) *)
phi2raw[7, 7, 0.9, 0.2] // N
0.696237 + 0.717812 i
Exp[-I kk[0.9, 0.2] 7]  // N
0.696237 + 0.717812 i
phi2raw[8, 8, 2, 3] // N
-0.549437 + 0.835535 i
Exp[-I kk[2, 3] 8]  // N
-0.549437 + 0.835535 i
```

```
(* Illustration: Geshkenbein Fig. 1 *)

f[k_] := Cos[k] + epsilon / (2 k) Sin[k]

f[k]
```

$$\text{Cos}[k] + \frac{5\,\text{Sin}[k]}{k}$$

```
plus[k_] := 1

minus[k_] := -1

epsilon := 10
 (* value for v=epsilon used in Geshkenbein Fig. 1 *)

pp1 = Plot[{f[k], plus[k], minus[k]},
  {k, 0, 19}, PlotRange → {{0, 19}, {-2, 2}},
  PlotStyle → RGBColor[1, 0, 0]]
```