A Method for series expansion of the bands of k-space Hamiltonians,v1

Anirudh Chandrasekaran

Loughborough University, United Kingdom

Last Updated: 26 July 2022.

Note: The first block of the code contains the definition of the function which, taking the Hamiltonian (as a function of k vector), the k point in question along with the values of tuning parameters, the band of interest, the maximum degree of the Taylor expansion, the k-space dimension, the maximum degree of expansion in the tuning parameters and the energy resolution as input, yields the Taylor expansion of the band of interest and other bands degenerate to it at the chosen k, to the chosen order. In the next section, we apply it to the Haldane model and compare the results to the Taylor expansion directly obtained from the dispersion (which is known).

In[ ]:=
```
(*This function is still in the works. It needs to
  be expanded to carefully incorporate the extension to non-
 trivially degenerate bands. In this version,
it can handle situations where the degenerate bands have same
 Taylor expansion up to the degree we are interested in.*)
TaylorExpandBand[Hamltn_, k0_, BandNo_, TaylorDegreeIn_,
   SpaceDimIn_ : -1, HoppingParamDegree_ : 1, Resln_ : 10^-9] :=
  Module[{Eigs, EigVals, EigVecs, HamltnDim, HamltnDerivs,
    EDerivs, TaylorPoly, TaylorPolyArray, HoppingDim, TaylorDegree,
    kVec, tVec, gn, fn, Degen, NonDegen, NoNonDegen, SpaceDim},
   If[Length[k0] > 0 && AllTrue[k0, NumberQ[#] &] &&
     IntegerQ[BandNo] && BandNo > 0 && IntegerQ[TaylorDegreeIn] &&
     TaylorDegreeIn > 0 && IntegerQ[HoppingParamDegree] &&
     HoppingParamDegree > 0 && IntegerQ[SpaceDimIn] && NumberQ[Resln]
    (*Some sanity checks to ensure that valid inputs have been fed in
      by the user. The point k0 is a vector containing both the k-
     point and the values of the tuning parameters,
    about which we will Taylor expand. i.e k0 = (k1,k2,...,kn,t1,t2,...,tm).*)
    ,
    If[SpaceDimIn == -1 || SpaceDimIn > Length[k0],
     (* If either the k-
      space dimension is not provided (in which case it defaults to -1),
     or if the provided value does not make sense when compared to
      dimension of k0, then we interpret the dimension of the k-
      point/vector of interest k0 to be the dimension of the full k-
      space (i.e there are no parameters t provided). *)
```

```
  SpaceDim = Length[k0];

  HoppingDim = 0; (* As a reflection of the statement above,
  none of the components in k0 represent hopping parameters,
  so that we set the dimension of the hopping sub-vector to zero. *)


  ,
  SpaceDim = SpaceDimIn;
  HoppingDim = Length[k0] - SpaceDimIn
  (* The number of hopping
    parameters is equal to dim k0 - dim of k-space. *)
];

TaylorDegree = TaylorDegreeIn + HoppingParamDegree;
(*This is needed since at the k order equal to TaylorDegree,
we would not have any terms involving the the tuning
   parameters t's if we did not expand to order TaylorDegree +
  HoppingParamDegree to begin with.*)

(*We now define symbolic k and t
  vectors which will be used for the series expansion.*)
kVec = Table[Symbol["p" <> ToString[i]], {i, 1, Length[k0]}];
tVec = Table[Symbol["δt" <> ToString[i]], {i, 1, HoppingDim}];

Eigs = Eigensystem[N[Hamltn[k0]]];
(* Computing the eigenvalues and eigenvectors at k0. *)

EigVals = Eigs[[1]];
EigVecs = Eigs[[2]];

(* For the calculations that will follow, we need the dimension of the
  Hamiltonian matrix, which is the same as the number of bands: *)
HamltnDim = Length[EigVals];

Eigs =
  SortBy[Table[Append[{EigVals[[i]]}, EigVecs[[i]]], {i, 1, HamltnDim}], First];
(* Sorting the eigensystem by the eigenvalues, rather than by their
  magnitudes, which is what Mathematica does by default. *)

EigVals = Eigs[[All, 1]];
EigVecs = Eigs[[All, 2]];
(* Separating the eigenvalues and eigenvectors in to separate lists *)

(* We now attempt to find the subset of eigenvectors that are degenerate
  to the band of interest, which is denoted by BandNo. Obviously,
the degeneracy is determined by the resolution adopted by us. *)
```

```
Degen = First@Last@Reap[Do[
      If[Abs[EigVals⟦i⟧ - EigVals⟦BandNo⟧] ≤ Resln, Sow[i]];
      , {i, 1, HamltnDim}]];

(* Having found the subset of bands dengenerate to the band of interest,
we now find the subset of bands that are not degenerate to it: *)
NonDegen = Complement[Table[i, {i, 1, HamltnDim}], Degen];


(* Number of bands that are not degenerate to BandNo: *)
NoNonDegen = Length[NonDegen];

(* Some sanity checks...can be removed. *)
Print["Bands to be series expanded: ",
 Degen, ", Rest of the bands: ", NonDegen];

(* Lists of matrices that we will
 be using for the Taylor expansion calculation: *)
HamltnDerivs = ConstantArray[0, TaylorDegree + 1];
EDerivs = ConstantArray[0, {TaylorDegree + 1, Length[Degen]}];
(* EDerivs contains the polynomials ∂^l ϵ_n that make up the Taylor
 expansion, stored order by order. It is a multidimensional
 list since we will be computing the Taylor expansion of each
 of the degenerate bands. For this version of the function,
this is a redundancy since we are assuming that the bands have
 the same Taylor expansion to the given order, not just at
 k0. This means that their Taylor expansions will be identical. *)

(* We now compute, order by order, the set of
 monomials that make up the Taylor expansion of the Hamiltonian,
all the way up to the order specified by TaylorDegree. For the
 convenience of the forthcoming calculations, it makes sense to
 transform all these into the basis of the eigenvectors at k0: *)
Do[HamltnDerivs⟦i + 1⟧ = Chop[Conjugate[EigVecs].
      (D[Hamltn[l * kVec + k0], {l, i}] /. {l → 0}).Transpose[EigVecs], Resln];
  , {i, 0, TaylorDegree}];
(* Normally we would do basis
 transformations of a matrix M by Evec†.M.Evec. However,
since Mathematica stores the eigenvectors in the form of rows,
we have to instead do (Evec)*.M.Evec^T. *)


(* The following is a function that will
   be needed in the calculation to follow. Basically it
   computes the quantity ⟨m_i|∂^(k_i-k_(i+1))H|m_(i+1)⟩-∂^(k_i-k_(i+1))ϵ_n.δ_(m_i,m_(i+1)) :*)
gn[mi_, ki_, mip1_, kip1_, n_] := If[
   1 ≤ mi ≤ HamltnDim && 1 ≤ mip1 ≤ HamltnDim && 1 ≤ kip1 < ki && MemberQ[Degen, n],
```

```
  HamltnDerivs⟦ki - kip1 + 1, mi, mip1⟧ - KroneckerDelta[mi, mip1]
    EDerivs⟦ki - kip1 + 1, First@First@Position[Degen, n]⟧, 0];
(* The +
  1 is needed in the array indices since Mathematica indexing begins from
   1 rather than 0. The first index then represents 0 in the conventional
   sense. Also we don't have to worry about the case when kip1 = ki,
since it will not occur. The function appears inside sums where
   kip1 runs from 1 to ki-1. This is the reason why we check kip1 <
 ki and not kip1 ≤ ki. Lastly, we use First@First@Position[Degen,n]
 instead of n directly since the EDerivs array is indexed by 1,...,
Length[Degen] and n instead represents the index of the band. *)

fn[ki_, mi_, n_] := If[MemberQ[Degen, n] && MemberQ[NonDegen, mi] && ki ≥ 1,
  1/(EigVals⟦n⟧ - EigVals⟦mi⟧) (HamltnDerivs⟦ki + 1, mi, n⟧ +
     Sum[Binomial[ki, kip1] Sum[gn[mi, ki, mip1, kip1, n] × fn[kip1, mip1, n],
        {mip1, NonDegen}], {kip1, 1, ki - 1}]), 0];


(*Having defined the functions necessary for our calculations,
we go ahead and compute the derivtive polynomials of the dispersion,
denoted by ∂ˡϵₙ, order by order:*)
EDerivs⟦1, All⟧ = ConstantArray[EigVals⟦BandNo⟧, Length[Degen]];


Do[
 Do[
   EDerivs⟦M + 1, First@First@Position[Degen, n]⟧ = FullSimplify[
      HamltnDerivs⟦M + 1, n, n⟧ + Sum[Binomial[M, k1] Sum[HamltnDerivs⟦
            M - k1 + 1, n, m1⟧ × fn[k1, m1, n], {m1, NonDegen}], {k1, 1, M - 1}]];
   , {M, 1, TaylorDegree}];
 , {n, Degen}];


(*Before processing the Taylor
 expansion and assembling it into a format we desire,
we first construct it using the polynomials calculated above.*)
TaylorPoly = ConstantArray[0, Length[Degen]];

Do[
 Do[
   TaylorPoly⟦n⟧ = TaylorPoly⟦n⟧ + Normal[Series[ReplaceAll[EDerivs⟦M + 1, n⟧/Factorial[M],
         Join[Table[kVec⟦i⟧ → λ kVec⟦i⟧, {i, 1, SpaceDim}], Table[
           kVec⟦j⟧ → γ tVec⟦j - SpaceDim⟧, {j, SpaceDim + 1, Length[k0]}]]],
```

```
          {γ, 0, HoppingParamDegree}]] /. {γ → 1};
    , {M, 0, TaylorDegree}];
  , {n, 1, Length[Degen]}];

(*Now we process the Taylor expansion into a nice and convenient form:*)
TaylorPolyArray = ConstantArray[0, {Length[Degen], TaylorDegreeIn + 1}];

Do[
 Do[
  TaylorPolyArray[[n, i + 1]] = 1/Factorial[i]
      Chop[D[TaylorPoly[[n]], {λ, i}] /. {λ → 0}, Resln] // FullSimplify;
  , {i, 0, TaylorDegreeIn}]
 , {n, 1, Length[Degen]}];

(*Time to return the Taylor expansion,
stored as a list, order by order in k:*)
TaylorPolyArray

,
Print[Style["Invalid inputs!", Red]]
]
];
```

We now benchmark the method using the Haldane model. To complicate things a bit, we double the orbitals making up the Hamiltonian of the Haldane model and reorder the basis a bit so that the method's ability to handle degenerate bands can be checked explicitly. We will tune the model slightly off the critical tuning of the staggered chemical potential M, which takes us very close to a monkey saddle $\left(k_x^3 - 3\, k_x\, k_y^2\right)$ at the K_- corner point of the Brillouin zone. We will then try to analyse the modulation of the the Taylor expansion, induced by the δM de-tuning correction to the staggered chemical potential.

```
In[ ]:= a1 = {1, 0};
a2 = {-1/2, Sqrt[3]/2};
a3 = {-1/2, -Sqrt[3]/2};
b1 = a2 - a3;
b2 = a3 - a1;
b3 = a1 - a2;

h2[k_] := 2 (Sin[k.b1] + Sin[k.b2] + Sin[k.b3]);

(*The diagonalized by hand,
dispersion of the Haldane model for the '+' band (that is band 2):*)
Ep[k_, t1_, t2_, m_] := ((m + t2 h2[k])^2 + t1^2 ((Cos[k.a1] + Cos[k.a2] + Cos[k.a3])^2 +
        (Sin[k.a1] + Sin[k.a2] + Sin[k.a3])^2))^(1/2);

(*The pristine two band Hamiltonian that we
 shall use to construct the four band Hamiltonian:*)
Hm[k_, t1_, t2_, MM_] = {{2 t2 (Sin[k.b1] + Sin[k.b2] + Sin[k.b3]) + MM,
    t1 ((Cos[k.a1] + Cos[k.a2] + Cos[k.a3]) + I (Sin[k.a1] + Sin[k.a2] + Sin[k.a3]))},
   {t1 ((Cos[k.a1] + Cos[k.a2] + Cos[k.a3]) - I (Sin[k.a1] + Sin[k.a2] + Sin[k.a3])),
    -2 t2 (Sin[k.b1] + Sin[k.b2] + Sin[k.b3]) - MM}};

(*A reordering of the basis orbitals:*)
NewBasis = {{1, 0, 0, 0}, {0, 0, 1, 0}, {0, 1, 0, 0}, {0, 0, 0, 1}};

(*Doubling the orbitals and rearranging them,
we obtain a four-band Hamiltonian:*)
Hamltn[pvec_] := Transpose[NewBasis].
  ArrayFlatten[IdentityMatrix[2] ⊗ Hm[{pvec[[1]], pvec[[2]]}, 1, 1/2, pvec[[3]]]].NewBasis
```

Note that the Hamiltonian defined by Hamltn[pvec] is a four band Hamiltonian that just yields two identical copies of the Haldane bands. As mentioned earlier, we are using this only to illustrate the fact that the method works for a narrow class of degenerate band situations, the scope of which will be expanded in the upcoming versions. We move on to compute the Taylor expansion to 4th order using the analytical dispersion. We don't use the usual Mathematica method for this since we want to make the order by order expansion containing the δM modulations explicit. To this end, we will store each order of the Taylor expansion as a separate element of a list.

In[ ]:=
```
TaylDegree = 4;(*Degree of Taylor
 polynomial. You can change this to any desired value.*)
MDegree = 2; (*Maximum power of the δM tuning terms to be included in the
 Tayloe expansion. This can also be changed to any desired value.*)

TaylPol = ConstantArray[0, TaylDegree + 1];
(*Some numerical choices of values for the parameters in the model:*)
t10 = 1;
t20 = 1/2;
M0 = (t10^2 - 18 t20^2)/(2 Sqrt[3] t20); (*Critical tuning.*)
Km = (4 π)/(3 Sqrt[3]) {0, 1};

(*Taylor expansion of the analytically
 diagnolised band. We compute it in this odd fashion,
rather than using Mathematica's inbuilt Series expansion function since
 we want to explicitly arrange the Taylor expansion order by order in k,
with the δM correction terms appearing to the
 requirted order in each of these terms.*)
Do[
 Do[
  TaylPol[[i + 1]] = TaylPol[[i + 1]] +
     (D[Ep[{λ p1, λ p2} + Km, t10, t20, -M0 + γ δM], {λ, i}, {γ, j}] /. {λ → 0, γ → 0} / (Factorial[i] Factorial[j]) //
       FullSimplify);
  , {j, 0, MDegree}];
 TaylPol[[i + 1]] = FullSimplify[TaylPol[[i + 1]]];
 , {i, 0, TaylDegree}]

TaylPol
```

Out[ ]=
$$\left\{\frac{1}{\sqrt{3}} - \delta M,\ 0,\ \frac{27}{8}\left(p1^2 + p2^2\right)\delta M\left(1 + \sqrt{3}\,\delta M\right),\ -\frac{3}{16}\,p2\left(-3\,p1^2 + p2^2\right)\left(4\sqrt{3} + 9\,\delta M\left(1 + \sqrt{3}\,\delta M\right)\right),\ \frac{81}{128}\left(p1^2 + p2^2\right)^2\left(3\sqrt{3} + \delta M\left(8 - \sqrt{3}\,\delta M\right)\right)\right\}$$

In order to compare with the algorithm, we convert the rational and symbolic numbers into numerical ones and expand all terms:

```
In[•]:= Expand[TaylPol] // N
```

Out[•]=
$$\{0.57735 - 1.\ \delta M,\ 0.,\ 3.375\ p1^2\ \delta M + 3.375\ p2^2\ \delta M + 5.84567\ p1^2\ \delta M^2 + 5.84567\ p2^2\ \delta M^2,\ 3.89711\ p1^2\ p2 - 1.29904\ p2^3 + 5.0625\ p1^2\ p2\ \delta M - 1.6875\ p2^3\ \delta M + 8.76851\ p1^2\ p2\ \delta M^2 - 2.92284\ p2^3\ \delta M^2,\ 3.28819\ p1^4 + 6.57638\ p1^2\ p2^2 + 3.28819\ p2^4 + 5.0625\ p1^4\ \delta M + 10.125\ p1^2\ p2^2\ \delta M + 5.0625\ p2^4\ \delta M - 1.09606\ p1^4\ \delta M^2 - 2.19213\ p1^2\ p2^2\ \delta M^2 - 1.09606\ p2^4\ \delta M^2\}$$

Thus we have the terms of the Taylor expansion arranged into a list, order by order. We can clearly see that there is a quadratic term and also other terms that are modulated by δM factors, which vanish at critical tuning (i.e at δM = 0). We will now attempt to reproduce this series expansion using our method. We can choose either band 3 or band 4 (they are identical since we simply “doubled” the orbitals of freedom of the original Hamiltonian and rearranged the basis. So we will get two identical copies of both the bands of the pristine Haldane model).

```
In[•]:= TaylPoly2 =
          TaylorExpandBand[Hamltn, Join[Km, {-M0}] // N, 4, TaylDegree, 2, MDegree][[1]];
        (*We choose [[1]] here since both [[1]] and [[2]] are identical
         Taylor expansions and will correspond respectively to bands 3 and 4*)

        Bands to be series expanded: {3, 4}, Rest of the bands: {1, 2}
```

```
In[•]:= Expand[TaylPoly2] /. {δt1 → δM} (*The method gives tuning parameters as δt1,
         δt2,..., δtn. So we have to replace them with
          the appropriate symbols in any given context.*)
```

Out[•]=
$$\{0.57735 - 1.\ \delta M,\ 0,\ 3.375\ p1^2\ \delta M + 3.375\ p2^2\ \delta M + 5.84567\ p1^2\ \delta M^2 + 5.84567\ p2^2\ \delta M^2,\ 3.89711\ p1^2\ p2 - 1.29904\ p2^3 + 5.0625\ p1^2\ p2\ \delta M - 1.6875\ p2^3\ \delta M + 8.76851\ p1^2\ p2\ \delta M^2 - 2.92284\ p2^3\ \delta M^2,\ 3.28819\ p1^4 + 6.57638\ p1^2\ p2^2 + 3.28819\ p2^4 + 5.0625\ p1^4\ \delta M + 10.125\ p1^2\ p2^2\ \delta M + 5.0625\ p2^4\ \delta M - 1.09606\ p1^4\ \delta M^2 - 2.19213\ p1^2\ p2^2\ \delta M^2 - 1.09606\ p2^4\ \delta M^2\}$$

We can check explicitly that this agrees exactly with the Taylor expansion obtained directly from the analytic dispersion. Try changing both TaylDegree and MDegree to different values and obtain the Taylor expansion by both the methods to order TaylDegree, containing terms modulated by different powers of δM, up to $\delta M^{\mathrm{MDegree}}$. Lastly, for the sake of completion, we can look at the Taylor polynomial at critical tuning (i.e δM = 0):

```
In[•]:= TaylPolyFull = 0;
        Do[
         TaylPolyFull += Expand[Poly] /. {δt1 → 0}
         , {Poly, TaylPoly2}]
        TaylPolyFull
```

Out[•]=
$$0.57735 + 3.28819\ p1^4 + 3.89711\ p1^2\ p2 + 6.57638\ p1^2\ p2^2 - 1.29904\ p2^3 + 3.28819\ p2^4$$

In the notebook ‘diagnosing_HOS.nb’, we explicitly check this polynomial for HOS and confirm that it has a monkey saddle.