CoulombHiggs.m v1.0

Jan Manschot, Boris Pioline, Ashoke Sen

February 21, 2013

The Mathematica package `CoulombHiggs.m` allows to compute the Poincar-Laurent polynomial of the moduli space of stable representations of quivers using the *Coulomb branch* and *Higgs branch* formulae. The latter is based on Reineke's solution to the Harder-Narasimhan recursion [1] and applies to quivers without oriented closed loops, while the former is based on a physical picture of BPS states as bound states of elementary 'single-centered' constitutents, and applies to any quivers with or without oriented loops [2–4]. This package was released together with the preprint [5] where a general algorithm for computing the index of the quantum mechanics of multi-centered BPS black holes (the Coulomb index) was outlined. The package file `CoulombHiggs.m` and example files `Kronecker.nb`, `Threenode.nb`,`Multinode.nb` and `CoulombIndexCheck.nb` were included in the "source" of this paper available in the arXiv submission. Uptodate versions can be obtained from the second named author's webpage,

`http://www.lpthe.jussieu.fr/~pioline/computing.html`

1 Basic usage

Assuming that the file `CoulombHiggs.m` is present in the user's Mathematica Application directory, the package is loaded by entering

```
In[1]:=   <<CoulombHiggs`
Out[1]:=  CoulombHiggs v 1.0 - A package for evaluating quiver
          invariants using the Coulomb and Higgs branch formulae.
```

If the file `CoulombHiggs.m` has not yet been copied in the user's Mathematica Application directory but is in the same directory as the notebook, evaluate instead

```
In[1]:=   SetDirectory[NotebookDirectory[]]; <<CoulombHiggs‘
Out[1]:=  CoulombHiggs v 1.0 - A package for evaluating quiver
          invariants using the Coulomb and Higgs branch formulae.
```

The first main routine is `CoulombBranchFormula`, whose basic usage is illustrated below:

```
In[1]:=   Simplify[CoulombBranchFormula[4{{0, 1, -1},{-1, 0, 1}, {1,
          -1, 0}}, {1/2, 1/6, -2/3}, {1, 1, 1}, y]]
Out[1]:=
```

$$2+\frac{1}{y^2}+y^2+\texttt{OmS}(\{1,1,1\})$$

The first argument corresponds to the matrix of DSZ products α_{ij} (an antisymmetric matrix of integers), the second to the FI parameters ζ_i (a vector of rational numbers), the third to the dimension vector N_i (a vector of integers), and the last to the angular momentum fugacity (a formal variable). In this example, the routine expresses the Poincaré-Laurent polynomial of the moduli space of a three-node Abelian cyclic quiver with 4 arrows between each subsequent node, in terms of the single centered degeneracy $\Omega^{\mathrm{S}}(\gamma_1+\gamma_2+\gamma_3)$.

The second main routine is `HiggsBranchFormula`, which computes the Poincaré-Laurent polynomial using the Higgs branch formula (**??**). The arguments are the same as for `CoulombBranchFormula`, but it only applies to quivers without oriented loop (although no test is made to ensure that this is the case):

```
In[1]:=   Simplify[HiggsBranchFormula[{{0, 3},{-3, 0}}, {1/2,-1/2},
          {2, 2}, y]]
Out[1]:=
```

$$-\frac{\left(y^2+1\right)\left(y^8+y^4+1\right)}{y^5}$$

The above command computes the Poincaré-Laurent polynomial for the Kronecker quiver with 3 arrows, FI parameters $(1/2,-1/2)$, dimension vector $(2,2)$. The package allows for much more, as documented below. Inline documentation can be obtained by typing e.g.

```
In[1]:=   ?CoulombBranchFormula
Out[1]:=
```

2 Symbols

- y: Angular momentum fugacity;
- `Om[`*charge vector_,y_*`]`: denotes the refined index $\Omega(\gamma, y)$;
- `Omb[`*charge vector_,y_*`]`: denotes the rational refined index $\bar{\Omega}(\gamma, y)$;
- `OmS[`*charge vector_,y_*`]`: denotes the single centered degeneracy $\Omega^{\rm S}(\gamma, y)$. The dependence on y can be omitted, but only at the end of the computation;
- `OmT[`*charge vector_,y_*`]`: denotes the (unevaluated) function $\Omega_{\rm tot}(\gamma, y)$
- `Coulombg[`*list of charge vectors_,y_*`]`: denotes the (unevaluated) Coulomb index $g_{\rm Coulomb}(\{\alpha_i\}, \{c_i\}, y)$, leaving the FI parameters unspecified;
- `HiggsG[`*charge vector_,y_*`]`: denotes the (unevaluated) stack invariant $G_{\rm Higgs}(\gamma, y)$;
- `CoulombH[`*list of charge vectors_,multiplicity vector_,y_*`]`: denotes the (unevaluated) factor $H(\{\alpha_i\}, \{n_i\}, y)$ appearing in the formula (5.2) for $\Omega_{\rm tot}(\sum n_i\alpha_i, y)$ in terms of $\Omega^{\rm S}(\alpha_i, y)$.
- `QFact[`*n_,y_*`]`: represents the (non-evaluated) q-deformed factorial $[n, y]!$

3 Environment variables

- `$QuiverPerturb1`: Sets the size of the perturbation $\epsilon_1 = 1/$`$QuiverPerturb` of the DSZ products, set to 1000 by default.
- `$QuiverPerturb2`: Sets the size of the perturbation $\epsilon_2 = 1/$`$DSZPerturb` of the DSZ products, set to 100000 by default.
- `$QuiverNoLoop`: If set to True, the quiver will be assumed to have no oriented loop, hence all H factors and all $\Omega^{\rm S}(\alpha)$ will be set to zero (unless α is a basis vector). Set to False by default.
- `$QuiverTestLoop`: If set to True, all H factors and $\Omega^{\rm S}(\alpha)$ corresponding to subquivers without loops will be set to zero (unless α is a basis vector). Set to True by default. Determining whether a subquiver has loops is time-consuming, so for large quivers it may be advisable to disable this feature. Note that `$QuiverNoLoop` takes precedence over this variable.

- `$QuiverMultiplier`: Overall scaling factor of the DSZ matrix in any evaluation of `Coulombg` or `HiggsG`. Set to 1 by default, could be a formal variable.
- `$QuiverVerbose`: If set to False, all consistency tests on data and corresponding error messages will be skipped. Set to True by default.
- `$QuiverDisplayCoulombH`: If set to True, the routine `CoulombBranchFormula` will return a list {`Q`, `R`} where `Q` is the Poincaré-Laurent polynomial and *R* is a list of replacement rules for the *CoulombH* factors. Set to False by default.
- `$QuiverPrecision`: Sets the numerical precision with which all consistency tests are carried out. This is set to 0 by default since all data are assumed to be rational numbers. This can be set to a small real number when using real data, however the user is warned that rounding errors tend to grow quickly.

4 Coulomb index

- `CoulombF[`*Mat_*,*Cvec_*`]`: returns the index of collinear solutions $F(\{\tilde\alpha_1,\cdots\tilde\alpha_n\},\{\tilde c_1,\cdots\tilde c_n\})$ with DSZ products $\tilde\alpha_{ij}=$ `Mat`$[[i,j]]$, FI terms $\tilde c_i=$ `Cvec`$[[i]]$ and trivial ordering.
- `CoulombG[`*Mat_*`]`: returns the index of scaling collinear solutions $G(\{\hat\alpha_1,\cdots\hat\alpha_n\})$ with DSZ products $\hat\alpha_{ij}=$ `Mat`$[[i,j]]$ and trivial ordering. The total angular momentum $\sum_{i<j}Mat[[i,j]]$ must vanish;
- `CoulombIndex[`*Mat_*,*PMat_*,*Cvec_*,*y_*`]`: evaluates the Coulomb index $g_{\rm Coulomb}(\{\alpha_1,\cdots\alpha_n\};\{c_1,\cdots c_n\};y)$ with DSZ products $\alpha_{ij}=$ `Mat`$[[i,j]]$, perturbed to *PMat*[[i,j]] so as to lift accidental degeneracies, possibly rescaled by an overall factor of `$QuiverMultiplier`, FI terms $c_i=$ `Cvec`$[[i]]$, angular momentum fugacity *y*;
- `CoulombFNum[`*Mat_*`]`: computes numerically the index $F(\{\tilde\alpha_1,\ldots\tilde\alpha_n\},\{\tilde c_1,\ldots\tilde c_n\})$ with DSZ matrix $\tilde\alpha_{ij}=$ `Mat`$[[i,j]]$ and FI parameters $\tilde c_i=$ `Cvec`$[[i]]$. Works only for few centers.
- `CoulombGNum[`*Mat_*`]`: computes numerically the scaling index $G(\hat\alpha_1,\ldots\hat\alpha_n)$ with DSZ matrix $\hat\alpha_{ij}=$ `Mat`$[[i,j]]$. Works only for few centers.
- `CoulombIndexNum[`*Mat_*,*PMat_*,*Cvec_*,*k_*,*y_*`]`: returns the Coulomb index $g_{\rm Coulomb}(\{\alpha_1,\cdots\alpha_n\};\{c_1,\cdots c_n\};y)$ with DSZ products $\alpha_{ij}=$ `Mat`$[[i,j]]$, possibly rescaled by an overall factor of `$QuiverMultiplier`, FI terms $c_i=$ `Cvec`$[[i]]$, angular momentum fugacity *y*, by searching collinear solutions numerically;

5 Coulomb branch formula

- `CoulombBranchFormula[Mat_,Cvec_,Nvec_,y_]`: computes is the Poincaré-Laurent polynomial of a quiver with DSZ products $\alpha_{ij}=$ `Mat`$[[i,j]]$ (possibly rescaled by `$QuiverMultiplier`) dimension vector N_i = `Nvec`$[[i]]$, FI parameters $\zeta_i=$ `Cvec`$[[i]]$, in terms of single-centered invariants $\Omega^{\rm S}$. This standalone routine first constructs the Poincaré-Laurent polynomial using (5.1), evaluates the Coulomb indices $g_{\rm Coulomb}$, and determines the H factors recursively using the minimal modification hypothesis. If `$QuiverDisplayCoulombH` is set to True, the routine returns a list {`Q`, `R`}, where `Q` is the Poincaré polynomial and *R* is a list of replacement rules for the *CoulombH* factors. For quivers without loops, the process can be sped up greatly by setting `$QuiverNoLoop` to True. For complicated quivers it is advisable to implement the Coulomb branch formula step by step, using the more elementary routines described below.

- `CoulombBranchFormulaFromH[Mat_,Cvec_,Nvec_,R_,y_]`: returns the Poincaré-Laurent polynomial of a quiver with DSZ products $\alpha_{ij}=$ `Mat`$[[i,j]]$, possibly rescaled by `$QuiverMultiplier`, dimension vector $N_i=$ `Nvec`$[[i]]$, FI parameters $\zeta_i=$ `Cvec`$[[i]]$, using the rule *R* to replace all *CoulombH* factors.

- `QuiverPoincarePolynomial[Nvec_,y_]`: constructs the Poincaré-Laurent polynomial of a quiver according to

$$\begin{aligned}
Q_{\rm Coulomb}(\gamma;\zeta;y) &= \sum_{m|\gamma}\frac{\mu(m)}{m}\,\frac{y-y^{-1}}{y^m-y^{-m}}\,\bar Q_{\rm Coulomb}(\gamma/m;\zeta;y^m)\\
\bar Q_{\rm Coulomb}(\gamma;\zeta;y) &= \sum_{n\geq 1}\sum_{\substack{\{\alpha_i\in\Gamma\}\\ \sum_{i=1}^n\alpha_i=\gamma}}\frac{g_{\rm Coulomb}\left(\{\alpha_1,\cdots,\alpha_n\},\{c_1,\cdots c_n\};y\right)}{|{\rm Aut}(\{\alpha_1,\cdots,\alpha_n\})|}\\
&\qquad\prod_{i=1}^n\left\{\sum_{\substack{m_i\in\mathbb{Z}\\ m_i|\alpha_i}}\frac{1}{m_i}\,\frac{y-y^{-1}}{y^{m_i}-y^{-m_i}}\,\Omega_{\rm tot}(\alpha_i/m_i;y^{m_i})\right\},
\end{aligned}\tag{5.1}$$

  Coincides with `QuiverPoincarePolynomialRat` for primitive dimension vector;

- `QuiverPoincarePolynomialRat[Nvec_,y_]`: constructs the rational Poincaré-Laurent polynomial $\bar Q_{\rm Coulomb}(\gamma;\zeta;y)$ according to the second line in (5.1);

- QuiverPoincarePolynomialExpand[*Mat_*,*PMat_*,*Cvec_*, *Nvec_*, *Q_*]: evaluates the Coulomb indices $g_{\rm Coulomb}$ and the total single center degeneracies $\Omega_{\rm tot}(\alpha_i,y)$ appearing in the Poincaré-Laurent polynomial Q of a quiver with DSZ products α_{ij} = Mat$[[i,j]]$, perturbed to PMat$[[i,j]]$ and rescaled by $QuiverMultiplier, dimension vector N_i = Nvec$[[i]]$, FI parameters ζ_i = Cvec$[[i]]$, using

$$\Omega_{\rm tot}(\alpha;y) = \Omega^{\rm S}(\alpha;y) + \sum_{\substack{\{\beta_i\in\Gamma\},\{m_i\in\mathbb{Z}\}\\ m_i\geq 1,\, \sum_i m_i\beta_i=\alpha}} H(\{\beta_i\};\{m_i\};y)\,\prod_i \Omega^{\rm S}(\beta_i;y^{m_i})\,. \quad (5.2)$$

- CoulombHSubQuivers[*Mat_*,*PMat_*,*Nvec_*,*y_*]:computes recursively all *CoulombH* factors for DSZ matrix *Mat*, perturbed to *PMat*, and any dimension vector strictly less than *Nvec*; relies on the next two routines:

- ListCoulombH[*Nvec_*,*Q_*]: returns returns a pair {ListH, ListC} where *ListH* is a list of *CoulombH* factors possibly appearing in the Poincaré-Laurent polynomial Q of a quiver with dimension vector *Nvec*, and *ListC* is the list of coefficients which multiply the monomials in $\Omega^{\rm S}(\alpha_i,y)$ canonically associated to the H factors in Q.

- SolveCoulombH[*ListH_*,*ListC_*, *rule_*]: returns a list of replacement rules for the *CoulombH* factors listed *ListH*, by applying the minimal modification hypothesis to the coefficients listed in *ListC*. The last argument is a replacement rule for *CoulombH* factors associated to subquivers.

- MinimalModif[*f_*]:returns the symmetric Laurent polynomial which coincides with the Laurent expansion expansion of the symmetric rational function f at $y=0$, up to strictly positive powers of y. Here symmetric means invariant under $y\to 1/y$. In practice, MinimalModif[*f*] evaluates the contour integral in [4], Eq 2.9

$$\oint \frac{{\rm d}u}{2\pi {\rm i}}\, \frac{(1/u-u)\,f(u)}{(1-uy)(1-u/y)} \quad (5.3)$$

by deforming the contour around 0 into a sum of counters over all poles of $f(u)$ and zeros of $(1-uy)(1-u/y)$. This trick allows to compute (5.3) even if the order of the pole of $f(y)$ at $y=0$ is unknown, which happens if $QuiverMultiplier is a formal variable.

- SymplifyOmSbasis[*f_*]:replaces $\Omega^{\rm S}(\gamma,y)\to 1$ when γ is a basis vector;

- SymplifyOmSbasismult[*f*_]: replaces $\Omega^{\rm S}(\gamma,y)\to 0$ when γ is a non-trivial multiple of a basis vector;
- CoulombHNoLoopToZero[*Mat*_,*f*_]: sets to zero any H factor in *f* corresponding to subquivers without loop, assuming DSZ products $\alpha_{ij}=$ Mat$[[i,j]]$; active only on 2-node subquivers if $QuiverTestLoop is set to False
- OmTNoLoopToZero[*Mat*_,*f*_]: sets to zero any $\Omega_{\rm tot}$ factor in *f* corresponding to subquivers without loop, assuming DSZ products $\alpha_{ij}=$ Mat$[[i,j]]$; active only on 2-node subquivers if $QuiverTestLoop is set to False
- OmSNoLoopToZero[*Mat*_,*f*_]: sets to zero any $\Omega^{\rm S}$ factor in *f* corresponding to subquivers without loop, assuming DSZ products $\alpha_{ij}=$ Mat$[[i,j]]$; active only on 2-node subquivers if $QuiverTestLoop is set to False
- EvalCoulombH3[*Mat*_,*f*_]: evaluates any 3-center H factor with multiplicity vector $\{1,1,1\}$ appearing in f. Not used in any routine so far.
- DropFugacity[*f*_]: drops second argument in any $\Omega^{\rm S}(\gamma,y)$ appearing in *f*, to be used at end of computation only;

6 Higgs branch formula

- HiggsBranchFormula[*Mat*_,*Cvec*_,*Nvec*_,*y*_]: computes the Poincaré-Laurent polynomial of a quiver with DSZ products $\alpha_{ij}=$ Mat$[[i,j]]$ (possibly rescaled by $QuiverMultiplier), dimension vector N_i = Nvec$[[i]]$, FI parameters ζ_i = Cvec$[[i]]$, using the Higgs branch formula. It is assumed, but not checked, that the quiver has no oriented closed loop;
- StackInvariant[*Mat*_,*Cvec*_,*Nvec*_,*y*_]: gives the stack invariant of a quiver with DSZ matrix $\alpha_{ij}=$ Mat$[[i,j]]$, possibly rescaled by an overall factor of $QuiverMultiplier, FI parameters ζ_i = Cvec$[[i]]$, dimension vector N_i = Nvec$[[i]]$, using Reineke's formula; the answer is written in terms of unevaluated q-deformed factorials *QFact[n,y]*;
- AbelianStackInvariant[*Mat*_,*Cvec*_,*y*_]: gives the Abelian stack invariant of a quiver with DSZ matrix $\alpha_{ij}=$ Mat$[[i,j]]$, possibly rescaled by an overall factor of $QuiverMultiplier, FI parameters ζ_i = Cvec$[[i]]$, using Reineke's formula; coincides with StackInvariant with *Nvec*= $\{1,\dots 1\}$ except that tests of marginal or threshold stability are performed (unless $QuiverVerbose is set to False);
- QDeformedFactorial[*n*_,*y*_]: gives the q-deformed factorial $[n,y]!$
- EvalQFact[*f*_]: evaluates any *QFact[n,y]* appearing in *f*

7 Utilities

- `ListAllPartitions[`*charge vector_*`]`:returns the list of unordered partitions $\{\alpha_i\}$ of the positive integer vector γ as a sum of positive, non-zero integer vectors α_i;

- `ListAllPartitionsMult[`*charge vector_*`]`:returns the list of unordered partitions $\{\alpha_i, m_i\}$ of the positive integer vector γ as a sum of positive, non-zero integer vectors α_i with multiplicity m_i;

- `ListSubQuivers[`*Nvec_*`]`:gives a list of all dimension vectors less or equal to *Nvec*;

- `SubDSZ[`*Mat_*,*Cvec_*,*Li_*`]`:gives the DSZ matrix of the subquiver made of vectors in list *Li*;

- `SymmetryFactor[`*Li_*`]`:gives the symmetry factor $1/|\mathrm{Aut}(\{\alpha_1, \alpha_2, \cdots, \alpha_n\}|$ for the list of charge vectors *Li*;

- `OmTRat[`*Nvec_*,*y_*`]`: gives the rational total invariant $\bar\Omega_{\rm tot}(\gamma;y)$ in terms of $\Omega_{\rm tot}(\gamma;y)$. Coincides with the latter if γ is primitive.

- `OmTToOmS[`*f_*`]`:expands out any $\Omega_{\rm tot}(\gamma;y)$ in f into H factors and $\Omega^{\rm S}$'s using(5.2);

- `OmToOmb[`*f_*`]`:expresses out any $\Omega(\gamma;y)$ in f in terms of $\bar\Omega(\gamma;y)$'s;

- `OmbToOm[`*f_*`]`:expresses out any $\bar\Omega(\gamma;y)$ in f in terms of $\Omega(\gamma;y)$'s;

- `HiggsGToOmb[`*Nvec_*,*y_*`]`:Returns the (unevaluated) HN invariant $G_{\rm Higgs}(\gamma,y)$ in terms of the rational refined indices $\Omega(\gamma;y)$ using (**??**);

- `OmbToHiggsG[`*Nvec_*,*y_*`]`:Returns the (unevaluated) rational refined index $\Omega(\gamma;y)$ in terms of the (unevaluated) stack invariants $G_{\rm Higgs}(\gamma,y)$ using (**??**);

- `RandomCvec[`*Nvec_*`]`:generates a random set of FI parameters ζ_i between -1 and 1, such that $\sum\zeta_i$ `Nvec`$[[i]]=0$;

- `UnitStepWarn[`*x_*`]`:gives 1 for $x>0$, 0 for $x<0$, and 1/2 if $x=0$. Produces a warning in this latter case, irrespective of the value of `$QuiverVerbose`. If so, run the computation again with a different random perturbation.

References

[1] M. Reineke, "The Harder-Narasimhan system in quantum groups and cohomology of quiver moduli," *Invent. Math.* **152** (2003), no. 2, 349–368.

[2] J. Manschot, B. Pioline and A. Sen, "Wall Crossing from Boltzmann Black Hole Halos," JHEP **1107**, 059 (2011) [arXiv:1011.1258 [hep-th]].

[3] J. Manschot, B. Pioline and A. Sen, "A Fixed point formula for the index of multi-centered N=2 black holes," JHEP **1105**, 057 (2011) [arXiv:1103.1887 [hep-th]].

[4] J. Manschot, B. Pioline and A. Sen, "From Black Holes to Quivers," JHEP **1211** (2012) 023 [arXiv:1207.2230 [hep-th]].

[5] J. Manschot, B. Pioline and A. Sen, "On the Coulomb and Higgs branch formulae for multi-centered black holes and quiver invariants", [arXiv:1302.nnnn [hep-th]].