CoulombHiggs.m v2.1

Jan Manschot, Boris Pioline, Ashoke Sen

April 30, 2014

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]. The first version of 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 second version 2.0, released along with the preprint [6], allowed to compute the Dolbeault-Laurent polynomial, relax assumptions on single-centered indices for basis vectors, study the effect of generalized mutations, and more. The third version 2.1, released along with the review [7], has been optimized to speed up the evaluation of Coulomb indices.

The package file `CoulombHiggs.m` and various example files 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 2.0 - 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 2.1 - 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: [1]

```
In[1]:= Simplify[CoulombBranchFormula[4{{0, 1, -1},{-1, 0, 1}, {1,
        -1, 0}}, {1/2, 1/6, -2/3}, {1, 1, 1}]]
Out[1]= 2 + 1/y^2 + y^2 + OmS({1, 1, 1}, y, t)
```

This routine computes the Dolbeault-Laurent polynomial of the quiver moduli space, expressed in terms of the single-centered indices. 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). The variables y and t are fugacities conjugate to the sum of the Dolbeault degrees $p+q$ (i.e. the angular momentum) and to the difference of the Dolbeault degrees $p-q$, respectively. The Poincaré-Laurent polynomial is obtained by setting $t=1$. For generic superpotential, the single-centered indices $\Omega^{\rm S}(\gamma,y)\equiv\Omega^{\rm S}(\gamma,y,1)$ are conjectured to be independent of y. In the above example, the Dolbeault polynomial of the moduli space of a three-node Abelian cyclic quiver with 4 arrows between each subsequent node is expressed in terms of the single-centered index $\Omega^{\rm S}(\gamma_1+\gamma_2+\gamma_3,y,t)$. The second main routine is `HiggsBranchFormula`, which computes the Poincaré-Laurent polynomial using the Higgs branch formula (which is only valid for quivers without oriented loop, but the routine works irrespective of this assumption). The arguments are the same as for `CoulombBranchFormula`:

[1] Note the following changes in v2.0: the fugacity y is no longer a parameter of CoulombBranchFormula and QuiverBranchFormula, and the former computes the Dolbeault polynomial in terms of $\Omega^{\rm S}(\alpha_i,t)$, rather than expressing the Poincaré polynomial in terms of $\Omega^{\rm S}(\alpha_i)$. Starting in v2.1, if $\sum_i N_i\zeta_i$ does not vanish, rather than issuing an error message, a uniform translation is applied internally to the ζ_i's. Other changes are highlighted by margin notes below.

```
In[1]:= Simplify[HiggsBranchFormula[{{0, 3},{-3, 0}}, {1/2,-1/2},
        {2, 2}]]
Out[1]= -((y^2+1)(y^8+y^4+1))/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: fugacity conjugate to the sum of Dolbeault degrees $p + q$ (i.e. angular momentum);
- t: fugacity conjugate to the difference of Dolbeault degrees $p - q$;
- `Om[`*charge vector_*`,`*y_*`]`: denotes the refined index $\Omega(\gamma, y)$;
- `Omb[`*charge vector_*`,`*y_*`]`: denotes the rational refined index $\bar{\Omega}(\gamma, y)$;

New in v2.0:

- `OmS[`*charge vector_*`,`*y_*`,`*t_* `]`: denotes the single-centered index $\Omega^{\mathrm{S}}(\gamma, y, t)$.
- `OmS[`*charge vector_*`,`*y_*`]`: denotes $\Omega^{\mathrm{S}}(\gamma, y) \equiv \Omega^{\mathrm{S}}(\gamma, y, t = 1)$.
- `OmS[`*charge vector_*`]`: denotes $\Omega^{\mathrm{S}}(\gamma, y)$, under the assumption that it is independent of y (which is conjectured to be the case for generic superpotential)
- `OmT[`*charge vector_*`,`*y_*`]`: denotes the (unevaluated) function $\Omega_{\mathrm{tot}}(\gamma, y)$;
- `Coulombg[`*list of charge vectors_*`,`*y_*`]`:: denotes the (unevaluated) Coulomb index $g_{\mathrm{Coulomb}}(\{\alpha_i\}, \{c_i\}, y)$, leaving the FI parameters unspecified;
- `HiggsG[`*charge vector_*`,`*y_*`]`: denotes the (unevaluated) stack invariant $G_{\mathrm{Higgs}}(\gamma, y)$;
- `CoulombH[`*list of charge vectors_*`,`*multiplicity vector_*`,`*y_*`]`: denotes the (unevaluated) factor $H(\{\alpha_i\}, \{n_i\}, y)$ appearing in the formula for $\Omega_{\mathrm{tot}}(\sum n_i\alpha_i, y)$ in terms of $\Omega^{\mathrm{S}}(\alpha_i, y)$.
- `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 $\epsilon_2 = 1/$`$DSZPerturb` of the DSZ products, set to 10^{10} by default.
- `$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.
- `$QuiverPrecision`: Sets the numerical precision with which all consistency tests are carried out. This is set to 0 by default since all data are assumed to be rational numbers. This can be set to a small real number when using real data, however the user is warned that rounding errors tend to grow quickly.
- `$QuiverRecursion`: If set to 1 (default value), then the new recursion relations from [5, v2] are used for computing `CoulombF`; if set to 0 the recursion relation from [5, v1] is used instead.

New in v2.0:

- `$QuiverOmSbasis`: Set to 1 by default. If set to 0, the routines `SimplifyOmSbasis` and `SimplifyOmSbasismult` are deactivated, so that the assumption that basis vectors carry $\Omega^{\rm S}(\ell\gamma_i) = \delta_{\ell,1}$ is relaxed.

New in v2.1:

- `$QuiverOpt`: Set to 1 by default. If set to 0, the routines `CoulombF`, `CoulombG`, `CoulombIndex` will use the non-optimized code provided in version 2.0, otherwise they use the optimized code provided in version 2.1.

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 $g_{\text{Coulomb}}(\{\alpha_1, \cdots \alpha_n\}; \{c_1, \cdots c_n\}; y)$ 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`$[[i]]$. For testing purposes only, works for up to 5 centers.

- `CoulombGNum[`*Mat_*`]`: computes numerically the scaling index $G(\hat{\alpha}_1, \ldots \hat{\alpha}_n)$ with DSZ matrix $\hat{\alpha}_{ij} =$ `Mat`$[[i,j]]$. For testing purposes only, works for up to 6 centers.

- `CoulombIndexNum[`*Mat_*,*PMat_*,*Cvec_*,*k_*,*y_*`]`: returns the Coulomb index $g_{\text{Coulomb}}(\{\alpha_1, \cdots \alpha_n\}; \{c_1, \cdots c_n\}; y)$ 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; For testing purposes only, works for up to 5 centers.

5 Coulomb branch formula

- `CoulombBranchFormula[`*Mat_*,*Cvec_*,*Nvec_*`]`: computes the Dolbeault polynomial of a quiver with DSZ products $\alpha_{ij} =$ `Mat`$[[i,j]]$, dimension vector $N_i =$ `Nvec`$[[i]]$, FI parameters $\zeta_i =$ `Cvec`$[[i]]$, in terms of single-centered invariants Ω^{S}. This standalone routine first constructs the Poincaré-Laurent polynomial, evaluates the Coulomb indices g_{Coulomb}, determines the H factors recursively using the minimal modification hypothesis and finally replaces y by t in the argument of Ω^{S} to construct the Dolbeault polynomial. 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_*]: returns the Dolbeault polynomial of a quiver with DSZ products $\alpha_{ij} =$ Mat$[[i,j]]$, 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. Coincides with QuiverPoincarePolynomialRat for primitive dimension vector;

- QuiverPoincarePolynomialRat[*Nvec_*,*y_*]: constructs the rational Poincaré-Laurent polynomial $\bar{Q}_{\rm Coulomb}(\gamma;\zeta;y)$;

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

- 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_*, *R_*]: returns a list of replacement rules for the *CoulombH* factors listed in *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

$$\oint \frac{\mathrm{d}u}{2\pi \mathrm{i}} \frac{(1/u - u)\, f(u)}{(1 - uy)(1 - u/y)} \tag{5.1}$$

  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 (5.1) even if the order of the pole of $f(y)$ at $y = 0$ is unknown, which happens if `$QuiverMultiplier` is a formal variable.

- `SimplifyOmSbasis[`*f_*`]`: replaces $\Omega^{\mathrm{S}}(\gamma, y) \to 1$ when γ is a basis vector, unless `$QuiverOmSbasis` is set to 0;

New in v2.0:

- `SimplifyOmSbasismult[`*f_*`]`: replaces $\Omega^{\mathrm{S}}(\gamma, y) \to 0$ when γ is a non-trivial multiple of a basis vector, unless `$QuiverOmSbasis` is set to 0;

- `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 Ω_{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 Ω^{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.

New in v2.0:

- `DropFugacity[`*f_*`]`: replaces $\Omega^{\mathrm{S}}(\gamma, y^m, t^m)$ by $\Omega^{\mathrm{S}}(\gamma, t^m)$ everywhere in f
- `SwapFugacity[`*f_*`]`: replaces $\Omega^{\mathrm{S}}(\gamma, y^m)$ with $\Omega^{\mathrm{S}}(\gamma, y^m, t^m)$ everywhere in f

6 Higgs branch formula

- HiggsBranchFormula[*Mat_*,*Cvec_*,*Nvec_*]: computes the Poincaré-Laurent polynomial of a quiver with DSZ products α_{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 loop;

- StackInvariant[*Mat_*,*Cvec_*,*Nvec_*,*y_*]: gives the stack invariant of a quiver with DSZ matrix α_{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 α_{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;
- `OmToOmb[f_]`: expresses any $\Omega(\gamma; y)$ in f in terms of $\bar{\Omega}(\gamma; y)$'s;
- `OmbToOm[f_]`: expresses 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 $\Omega(\gamma; y)$;
- `OmbToHiggsG[Nvec_,y_]`: Returns the (unevaluated) rational refined index $\Omega(\gamma; y)$ in terms of the (unevaluated) stack invariants $G_{\rm Higgs}(\gamma, y)$;
- `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, the user is advised run the computation again with a different random perturbation. For efficiency, this instruction is no longer used in v2.1, however a warning is still issued if one encounters a Heaviside function with zero argument in the evaluation of the Coulomb indices.
- `GrassmannianPoincare[k_,n_,y_]`: computes the Poincaré polynomial of the Grassmannian $G(k, n)$ via Eq. (6.22) in [4].

New in v2.0:

- `CyclicQuiverOmS[avec_,t_]`: computes the single-centered index $\Omega^{\rm S}(\gamma_1, \ldots, \gamma_K)$ associated to a cyclic Abelian quivers with DSZ matrix $\alpha_{i,j+1} =$ `avec`$[[i, i+1]]$ via Eq (4.29) in [4].

New in v2.1:

- `QuiverPlot[Mat_]`: Displays the quiver with DSZ matrix *Mat*.
- `FIFromZ[Nvec_,Zvec_]`: Computes the FI parameters $\{c_i\}$ from the vector of central charges `Zvec` $= \{Z_i\}$ and dimension vector `Nvec` $= \{N_i\}$ via $c_i = \Im(e^{-i\phi} Z_i)$, where ϕ is the argument of $\sum_i N_i Z_i$. The parameters c_i are rounded up to the nearest rational number with denominator less than `$QuiverPerturb1`.

7.1 Mutations

New in v2.0:

The following routines and environment variables were introduced in `CoulombHiggs.m` v1.1, to allow investigation of mutations of generalized quivers [6]:

- `MutateRight[`*Mat_*,*Cvec_*,*Nvec_*,*k_*`]`: Computes the DSZ matrix, FI parameters and dimension vector of the quiver obtained by applying a right-mutation with respect to the node k. If k is a list $\{k_i\}$, then the right mutations k_i are applied successively, starting from the last entry in k. No consistency check on the FI parameters is performed.
- `MutateLeft[`*Mat_*,*Cvec_*,*Nvec_*,*k_*`]`: Computes the DSZ matrix, FI parameters and dimension vector of the quiver obtained by applying a left-mutation with respect to the node k. If k is a list $\{k_i\}$, then the right mutations k_i are applied successively, starting from the last entry in k. No consistency check on the FI parameters is performed.
- `OmStoOmS2[`*f_*`]`: replaces `OmS[gam, y, t]` by `OmS2[gam, y, t]` anywhere in `f`. This is useful for distinguishing the single-centered invariants of the mutated quiver from those of the original one.
- `MutateRightOmS[`*Mat_*,*k_*,*f_*`]`: expresses the single-centered invariants `OmS[gam, y, t]` of the original quiver with DSZ matrix *Mat* in terms of the single-centered invariants `OmS2[gam, y, t]` of the quiver obtained by right-mutation with respect to node k, using Eq. 1.13 in [6].
- `MutateLeftOmS[`*Mat_*,*k_*,*f_*`]`: expresses the single-centered invariants `OmS[gam, y, t]` of the original quiver with DSZ matrix *Mat* in terms of the single-centered invariants `OmS2[gam, y, t]` of the quiver obtained by left-mutation with respect to node k, using Eq. 1.13 in [6].
- `MutateRightOmS2[`*Mat_*,*k_*,*f_*`]`: expresses the single-centered invariants `OmS2[gam, y, t]` a quiver with DSZ matrix *Mat* in terms of the single-centered invariants `OmS[gam, y, t]` of the quiver obtained by right-mutation with respect to node k. Identical to `MutateRightOmS`, except for swapping `OmS[gam, y, t]` and `OmS2[gam, y, t]`.
- `MutateLefttOmS2[`*Mat_*,*k_*,*f_*`]`: expresses the single-centered invariants `OmS2[gam, y, t]` a quiver with DSZ matrix *Mat* in terms of the single-centered invariants `OmS[gam, y, t]` of the quiver obtained by right-mutation with respect to node k. Identical to `MutateLeftOmS`, except for swapping `OmS[gam, y, t]` and `OmS2[gam, y, t]`.
- `DropOmSNeg[`*f_*`]`: equates to 0 any $\Omega^{\rm S}(\gamma, y, t)$ where the dimension vector associated to γ has negative components.
- `$QuiverMutationMult`: Equal to 1 by default. Set to M, defined in Eq. (1.8) of [6] when dealing with generalized quivers.

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," JHEP **1305** (2013) 166 [arXiv:1302.5498 [hep-th]].

[6] J. Manschot, B. Pioline and A. Sen, "Generalized quiver mutations and single-centered indices," JHEP **1401** (2014) 050 [arXiv:1309.7053 [hep-th]].

[7] J. Manschot, B. Pioline and A. Sen, "The Coulomb branch formula for quiver invariants", [arXiv:1404.7154 [hep-th]].