NAME

surgen.in, — General input file for **surfgen**.

SYNOPSIS

This input file is used by coupled potential surface fitting program **surfgen** and its potential evaluation library. For a detailed description of the program and the evaluation library, see surfgen(1), potlib(1)

Place **surfgen.in** in the same directory where **surfgen** is being called.

DESCRIPTION

surgen.in is a Fortran 90 namelist input file. This file contains input parameters that controls the behavior of **surfgen**. This includes the **GENERAL** namelist input that specify the job type and, along with other input files, defines the molecular system, the electronic states, their symmetry, the internal coordinate basis and the expansion of Hd. Most job types will have another namelist that defines parameters that are specific to a job, except for *jobtype=-1*.

The evaluator library also use the general namelist to define the expansion. It also use a POTLIB namelist to define behaviors such as geometry logging and error estimation.

Developed by Yarkony group, Johns Hopkins University 2010-2013.

NAMELIST INPUT

GENERAL

General input that controls the expansion ansatz of the quasi-diabatic Hamiltonian (Hd), and the job to be performed by **surfgen**. Input files *connect.in*, *coord.in*, and *irrep.in* also contain detailed parameters about the definition of the ansatz.

List of Input Parameters

jobtype — **INTEGER [0]** Specifies the type of job to be performed. Currently the following Types are implemented:

- -1 Use the molecule and connectivity definitions to generate all feasible symmetry permutations and print them to the standard output.
- 0 Do nothing during execution. During initialization the program will construct the expansion and read the coefficients from Hd storage file. Used by potential evaluation libraries only. Requires namelist: *POTLIB*
- 1 Construct fit Hd from ab initio data. Required namelist: *MAKESURF*
- 2 Find minima or conical intersections on the ab initio surface corresponding to a fit Hd. Requires namelist: *MINMEX*

natoms — **INTEGER [0]** Number of atoms in the molecular system. It is usually unwise to use the default value of this parameter.

order — **INTEGER [2]** Maximum order of the Hd expansion. Order here means the number of basis functions that are multiplied together, should they be same type of coordinate or not, in the form of coordinates defined in *coord.in.*

atmgrp — **INTEGER,dimension(natoms)** Array that specify the equivalency of the atoms. Must contain *natoms* elements, each of which specify the group index of an atom. Atoms in the same group are considered equivalent by the program and the permutations among them will be generated as symmetry operations.

nGrp — **INTEGER** Number of electronic state groups. Each group contains a set of electronic states that carry an irreducible representation. For example, for OH radical the lowest 3 states are contained in 2 groups, an E group with 2 states and A group with 1 state.

groupsym — **INTEGER,dimension(ngrp)** Index of permutational irreducible representations carried by each of the state groups. For each irrep, representation matrices for every permutation must be supplied in input file irrep.in Irreps are indexed as they appear in *irrep.in*

groupprty — **INTEGER,dimension(ngrp)** Inversion symmetry (parity) of each of the state group.

printlvl **INTEGER [1]** Controls the level of infomation printed to standard output. 0 is lowest and 5 is highest level.

inputfl **CHARACTER(72) ['hd.data']** Name of Hd expansion coefficient input file. Hd will be initialized using these coefficients. When empty or file not exist, uncoupled surface with constant (but not identical) energies will be used to initialize Hd.

MAKESURF

This namelist contains parameters that control the surface fitting procedure. These are only used for *jobtype=1.*

Input File Specifications These options tell the program where to look for *ab initio* data. The data should be divided into groups of data points, and the information of each group is stored in a separate directory. The program will search for data in each of these directories to determine which piece of data is available, and append all available data to fitting set. This is a new functionality in 2.1 and is intended to allow the program to use results from calculations where not all data are obtained. For example, calculation of only energy, a fewer number of states, or where states are well separated and couplings are not calculated. It is also intended to make data storage more managable and transparent by grouping different data in separate directories.
The name of input files can be specified for each directory. The filenames of gradients and coupling data are expected to contain one and two wildcard characters

SearchPath **CHARACTER(72),dimension(100) ['.',99*'']** Paths of directories where the program should look for sets of input file. Each path should contain all the input information of a group of points. Including geometries, energies, and optionally energy gradients and couplings.

noteptn **CHARACTER(72),dimension(100) [100*'note']** Specifies the filename of the optional file where a brief note is kept to explain where the geometries in the current path are and what data are available. The first line of this file will also be printed to standard output when the program reads this directory. No wildcard allowed. Optional.

gmfptn **CHARACTER(72),dimension(100) [100*'geom.all']** Name of geometry input file. No wildcard characters allowed. Required.

enfptn **CHARACTER(72),dimension(100) [100*'energy.all']** Name of energy input file. No wildcard characters allowed. Required. If the file does not contain all the state, please add in the first line of the file
STATES *st1 st2*
where *st1* and *st2* are the lowest and highest state of which the energy is included in this file. For example, if the directory contains data from hessian calculation on state 3, then the line should be
STATES 3 3

grdfptn **CHARACTER(72),dimension(100) [100*'cartgrd.drt1.state$.all']** Pattern for energy gradient input file. Has 1 wildcard character which holds the slot for the index of the state of which the gradients are calculated. Optional. The program will search all the states that has an energy data in energy input file.

cpfptn **CHARACTER(72),dimension(100) [100*'cartgrd.drt1.state$.drt1.state$.all']** Pattern for derivative coupling input file. Has 2 wildcard characters which holds the slot for the indices of the pair of states between which the couplings are calculated. Optional. The program will search all the pairs of states that both has an energy data in energy input file.

Data Selection and Weighing

npoints **INTEGER [0]** Number of points to be fit. Note that if the program cannot find the specified number of points, the variable will be adjusted to the actual number of data points read from files. If there are more data than specified, the program will only use the first *npoints* data points.

eshift	**DOUBLE PRECISION [.0]** A uniform shift applied to all `ab initio` energies.
gcutoff	**DOUBLE PRECISION [1D-14]** The threshold below which gradients will be considered vanished and treated as exactly 0.
usefij	**LOGICAL [.true.]** Specifies if the derivative couplings instead of derivative coupling times energy differences will be used as coupling input. Derivative couplings approach infinity at intersections while coupling times energy difference remain well behaved everywhere.
w_energy	**DOUBLE PRECISION [1.]** Weight factor for energy equations. This factor is multiplied with point weights and high energy scaling weights to yield the final weight of equations.
w_grad	**DOUBLE PRECISION [1.]** Weight factor for energy gradient equations. This factor is multiplied with point weights and high energy scaling weights to yield the final weight of equations.
w_fij	**DOUBLE PRECISION [1.]** Weight factor for coupling equations. This factor is multiplied with point weights and high energy scaling weights to yield the final weight of equations.
energyT	**DOUBLE PRECISION,dimension(10) [1D30]**
highEScale	**DOUBLE PRECISION,dimension(10) [1.]** *energyT* specifies a series of thresholds for the downscaling of equations when the ab initio energy of an electronic state is very high. When *E>energyT(i)*, weight **highEScale(i)** is applied to the energy, gradient and derivative coupling equations that involve that state. For couplings, the higher state is used to determine the weight. The highest possible energy bracket (with lowest weight) will be used.
ediffcutoff	**DOUBLE PRECISION [20.]**
nrmediff	**DOUBLE PRECISION [2D4]** The weight for derivative coupling equations is weighed down by factor **nrmediff/(ΔE+ediffcutoff)**. This weighing procedure is due to the fact that coupling times energy difference is being fit instead of the coupling itself, which is singular near intersections. Increasing the weight according to energy difference ensures that residue couplings are properly minimized, and the cutoff term prevents problematic singular behavior. This prevents the mathematical complexity of directly taking deratives of the couplings with respect to fitting coefficients, which will give rise to term that correspond to change in energy difference.
ediffcutoff2	**DOUBLE PRECISION [1.]**
nrmediff2	**DOUBLE PRECISION [100.]** Similar to the above case, the energy equations are weighed up by factor **nrmediff2/(ΔE+ediffcutoff2)** if this value is greater than 1. This is to ensure that energy differences are properly reproduced for points that are close to degeneracy.

Fitting Algorithm and Acceleration

maxiter	**INTEGER [3]** Maximum number of iterations for the fitting algorithm.
toler	**DOUBLE PRECISION [1D-3]** Convergence tolerance for change in expansion coefficient.
maxd	**DOUBLE PRECISION [1D0]** Maximum allowed change in Hd expansion coefficients between iterations.
dfstart	**INTEGER [0]** Iteration at which differential convergence will be started. The normal equations will be constructed for the *change* of coefficients instead of expansion coefficients themselves. This will usually result in better fit and allows dumping while lifting the flattening term to very small value. However, this convergence mode has more tendency to experience oscillations and should not be enabled if the fit is qualitatively incorrect. It is recommended that when differential convergence is enabled, set *DijScale=1* and *DijScale2=1*
exactTol	**DOUBLE PRECISION [1D-12]** Eigenvalue cutoff when solving constrained normal equations. This parameter dictates how accurate the exact equations will be reproduced.
LSETol	**DOUBLE PRECISION [1D-7]** Diagonal shift on the normal equations when solving linear equations. Larger value leads to more stable but usually slower convergence.
flattening	**DOUBLE PRECISION [1D-8]** Flattening term that will be included in the objective function. In differential convergence mode, this option will remove contributions that have very small contributions to the quality of fit. As opposed to `LSETol`, which only changes the

convergence procedure but does not affect the converged results, `flattening` changes the Lagrangian and thus will result in a different converged Hd.

ndiis	**INTEGER [10]** Maximum dimensionality of DIIS interpolation space
ndstart	**INTEGER [10]** The number of iterations to start DIIS interpolation.
linSteps	**INTEGER [0]** Number of linear steps to perform. When greater than 0, the program will break the predicted change into **linSteps** smaller steps and try to find the step length that yields the smallest gradient for the Lagrangian. Step sizes are automatically shrinked when the norm of the gradient increases.
linNegSteps	**INTEGER [0]** Number of linear steps to be taken to the opposite direction of the predicted change but with the same size. This should only be used when the normal equations fail to give the correct direction of changes and the linear steps towards the positive direction encounter an immediate increase in the norm of Lagrangian.
DijScale	**DOUBLE PRECISION [1.]** This option controls the multiplier of the derivative of eigenvectors with respect to the fitting coefficients. When set to 0, the dependency of eigenvectors on fitting coefficients are ignored. When set to 1, the first order response of eigenvectors with respect to the change in fitting coefficients is fully implemented. This option is used by the construction of normal equations as well as evaluation of gradients of the Lagrange multipliers. It is recommended to have *DijScale=1.0* in most cases. It only needs to be turned down when eigenvectors are changing too rapidly and gives oscillations.
DijScale2	**DOUBLE PRECISION [1.]** Similar to **DijScale**, this option is an additional multiplier that only scales the eigenvector derivatives in the normal equations, but does not affect the evaluation of Lagrangian gradients.
scaleEx	**DOUBLE PRECISION [1.]** Uniformly scale all exact equations. Since there is no weight for exact equations, this is done through scaling the gradient of the Lagrangian with respect to Lagrange multipliers. This option normally does not need to be changed. Only use it when convergence problems occur.
stepMethod	**INTEGER [0]** This option specifies the algorithm used for fitting procedure. Currently *method 0* uses linear equality constrained least squares equations and *method 1* uses gradient projection technique. **Method 1 is not working very well at the moment. Please use 0.**
ExConv	**DOUBLE PRECISION [1D-5]** Convergence criteria for exact equations. Used by *stepMethod=1*.
maxED	**DOUBLE PRECISION [1D-2]** Maximum step length for exact step. Used by *stepMethod=1*.
mmiter	**INTEGER [10]** Number of micro iterations for *stepMethod=1*.
gscaler	**DOUBLE PRECISION [1D-5]** Scaling factor for projected gradient. Used by *stepMethod=1*.

Eigenvector Ordering and Phasing

enfDiab	**INTEGER [0]** Specify a point where diabatic and adiabatic representation will be forced to coincide. Every iteration the program will force the eigenvector of this point to be unit vectors. The off-diagonal element will be fit to 0 and the derivative of eigenvectors at this point (Dij) will also be 0 under all conditions. The adiabatic-diabatic transformation is subject to a globally constant transformation. Since such transformation does not affect the Hamiltonian in any manner, it cannot be determined from the fitting procedure itself. When states have different symmetry, such degree of freedom can be removed through the use of correct symmetry. When some states carry the same symmetry, this option is used to eliminate the extra degree of freedom.
gorder	**DOUBLE PRECISION [1D-3]** Threshold for energy difference below which the states will by ordered by gradients instead of absolute energy. This option is ignored when *followPrev=.true*.

ckl_input	**CHARACTER(72) ['']** Input file that contains the initial guess of eigenvectors at each data point. When left empty or file not exist, the eigenvectors are initialized by diagonalizing initial Hd.
ckl_output	**CHARACTER(72) ['ckl.out']** Output file that contains the final eigenvectors at each data point.
followPrev	**LOGICAL [.false.]** Whether the new eigenvectors will be ordered and phased to match the vectors from the previous iteration. This allows a more consistent and smoother convergence but may increase the tendency to match the states in a non-optimal way.
maxRot	**DOUBLE PRECISION [.0]** When set greater than 0, the eigenvectors rotation between rotations are monitored and when the rotation is larger than this parameter the rotation is dumped to this value. **HAVE NOT BEEN TESTED FOR MORE THAN 2 STATES. DO NOT USE IT IF YOU HAVE >2 STATES!**

Local Coordinate Construction

useIntGrad	**LOGICAL [.true.]** Specifies wether the gradients and derivative couplings will be fit using Cartesian components or a transformed coordinate constructed at each point that removes the null equations (translations, rotations, relative motion of dissociated fragments and symmetry zeros). This coordinate is constructed by obtaining the eigenvectors of matrix B^T.B, where B is the Wilson's B matrix.
intGradT	**DOUBLE PRECISION [1D-3]** Threshold for eigenvalue cut off of B^T.B matrix. When an eigenvalue is lower than *intGradT*, the coordinate is considered non-internal and removed from the fitting equaitons.
intGradS	**DOUBLE PRECISION [1D-1]** Threshold for diminished weights. New coordinates that correspond to eigenvalues lower than *intGradS* will be weighed by factor **ev/intGradS**, where **ev** is the eigenvalue.
gScaleMode	**INTEGER [2]** Controls how the gradients and couplings will be weighed according to *intGradS*. Available scaling methods are : =0 Do not scale; >0 Scale all coordinates; <0 Scale couplings only
deg_cap	**DOUBLE PRECISION [1D-5]** Threshold for energy difference below which the states will be considered degenerate. Intersection adapted coordinate will be used for these electronic states. *Degeneracy for more than 2 states is coded but never tested.*

Removal of Null Space

TBas	**DOUBLE PRECISION [1D-6]** Theshold for eigenvalue cutoff of the primitive basis overlap matrix. This controls the degree of linear dependency that will be allowed in the basis constructed for the fit.
ecutoff	**DOUBLE PRECISION [1.]** Energy threshold in *hartree* above which the energy data will not be considered in null space removal procedure. This is used to prevent the equations that are irrelevant from introducing extra degrees of freedom.
egcutoff	**DOUBLE PRECISION [0.6]** The gradients and couplings data of a point will not be considered in null space removal procedure when the ab initio energy of the lowest state is higher than this value. Similar to **ecutoff**, this parameter is used to prevent irrelevant high energy data points from introducing unnecessary degrees of freedom.

Input and Output

outputfl	**CHARACTER(72) ['']** Name of the output file that will store the fit surface.
flheader	**CHARACTER(72) ['----']** Header that will be printed into the description field of Hd storage file.
rmsexcl	**INTEGER [0]** This parameter controls if low weight points will be included in the RMS analysis. Points with weight lower than **-1/rmsexcl** will be excluded when *rmsexcl<0*. No effect when *rmsexcl>=0*

Testing

ntest	**INTEGER [0]** Number of test points. When greater than 0, Hd gradient test will be performed. Used only for debugging purpose.

POTLIB

Parameters that control the behavior of the potential evaluation library.
For the moment, please use default settings.

SEE ALSO

`connect.in`(1) `coords.in`(1), `irrep.in`(1), `hd.data`(1), `points.in`(1), `potlib`(1), `surfgen`(1),

BUGS

Please send bug reports to Xiaolei Zhu ⟨virtualzx@gmail.com⟩