6 Emulating Principal Components

This section is incomplete, and the PCA software is not yet fully tested.

6.1 Summary

Rather than emulating all observables, it can be more efficient to emulate a handful of principal components. After generating the training-point data, one can run the `pca` program included with the distribution. This will create files that shadow those used to emulate the observables. This will create a file `Info/pca_info.txt` alongside `Info/observable_info.txt`. The difference is that the observables will be named `z1,z2···`. In each run directory, alongside the `obs.txt` files, there will be a `obs_pca.txt` file. Finally, there will be a file `PCA_Info/tranformation_info.txt` file that contains all the information and matrices required to perform the basis transformation. If the parameter `Use_PCA` is set to `true`, the emulator will use the PCA files above instead of the observable files. The emulator will then store the Taylor coefficients in the directory `coefficients_pca/` rather than in `coefficients/`.

To get an idea of the capabilities and functionality of the PCA elements of the Smooth Emulator Distribution, one can view the sample main program,
`GITHOME_MSU/smooth/local/main_programs/pca_main.cc`.

6.2 PCA Parameters (not model parameters!)

The PCA programs uses parameter that are prefixed with **SmoothEmulator**. One would typically use the same parameter file as used for running Smooth Emulator. The relevant parameters are:

1. **SmoothEmultor_UsePCA**
   If one wishes to emulate the PCA observables, i.e. those that are linear combinations of the real observables, this should be set to true. One must then be sure to have run the PCA decomposition programs first.
2. **SmoothEmulator_ModelRunDirName** and **SmoothEmulator_TrainingPts** should be set the same as used by Smooth Emulator.

6.3 Running the PCA programs

The first sample program is `pca_calctransformation`, which reads the training information from the full model runs and calculates the principal component information. Quantities such as the PCA eigenvalues and eigenvectors are stored. This provides the User knowledge of which linear combination of observables carry significant resolving power. By storing the eigenvectors, the information may be retrieved later. This allows the User to easily transform from the observable, $\boldsymbol{y_a}$, to the PCA components $\boldsymbol{z_a}$.

One should first to `GITHOME_MSU/local/build` and compile/install the program
`GITHOME_MSU/local/bin/pca_calctransformation`.

```
.../local/build % cmake .
```

```
.../local/build % make pca_calctransformation
```

Next, from the project directory (assuming the training point information has already been collected) one can enter the command (assuming the path includes `GITHOME_MSU/local/bin`)

```
../my_project/ % pca_calctransformation PARAMETER_FILENAME
```

Here, `PARAMETER_FILENAME` is likely `parameters/emulator_parameters.txt`. At this point, all the information about observables from the training has equivalent representation for the PCA components.

In order to emulate the PCA components, one must set the parameter `SmoothEmulator_UsePCA` to true. Then, running the program `smoothy_tune` as described above will build and tune an emulator for the PCA components. It will store the Taylor coefficients in the directory `coefficients_pca/`.

The second sample program reads the transformation information written by `pca_calctransformation`. This program gives an example of transforming a vector of principal components, $\boldsymbol{z_a}$, to a vector of observables $\boldsymbol{y_a}$. To compile the programs, change into the build directory as above and enter:

```
.../local/build % cmake .
.../local/build % make pca_readtransformation
```