Overrider Class

Fields

Input/Output methods

```
overrider();
enum typeData {NODATA, SPLINE, FUNCTION};

int nbIndices;                                  <-- void extend(int newNbIndices);
vector<typeData> typeStorage;                   <-- void reset();
vector<tk::spline*> dataSpl;                    --> bool hasData(int index);
vector<double (*)(double)> dataFunct;           <-- void learnSpl(int index, vector<double> xs, vector<double> ys, bool yesSplnoLinearInterpol);
                                                <-- void learnFunct(int index, double (*f)(double));
                                                --> double operator ()(int index, double value);
                                                    -> returns interpolated value of curve index at position value
vector<bool> override;                          <-- void setOver(int index = -1, bool value = true);
    -> variables/indexes to override            --> bool operator ()(int index);
                                                    string print();
```

TableCourse Class (Kinetic Data)

Fields

Input/Output methods

```
int nbVar;                                      <-- TableCourse(int _nbVar);
int nbLignes;                                   <-- void setHeader(int i, string title);
    vector<string> headers;
    vector<double> attribut;
    vector< vector<double> * > storage;         <-- void addSet(double attr, vector<double> &toCopy);
                                                    TableCourse(string fileToRead);
                                                    TableCourse(TableCourse* toCopy);
                                                --> void read(string fileName);
                                                    void save(string fileName, string title = string(""));
                                                    vector<double> getTimeCourse(int var);
                                                    vector<double> getTimePoints();
                                                    double operator()(int vari, typeTime timej);
                                                    TableCourse subKinetics(vector<int> timePoints, vector<string> namesVariables);
                                                    void reset();
                                                    string print(bool fileExportVersion = true);
```

KineticData File

```
nbRows(time-pts)    nbColumns(nbVars)
headerLeft    headerVar1    headerVar2    ...
time1         valVar1       valVar2       ...
time2         valVar1       valVar2       ...
...
```

Evaluator Class (point-by-point database)

Fields

Input/Output methods

```
                int nbPoints;
list of variables
and time-points vector<int>          Species;
                vector<typeTime>     Times;
experimental
data (optional) vector<double>       ExpectedValues;
                vector<double>       StdDeviations;
Wanted data
from simulations vector<double>      theData;
                vector<bool>         recorded;
                bool recording_before_simulation;
```

```
double getVal(int _Species, typeTime _time_sec, double _expectedValue = 0, double _stdDev = 0);

typeTime nextPoint();

bool takeDataAtThisTime(typeTime t_sec);

int speciesToUpdate();
void setValNow(int _Species, double value);

void resetDataKeepingExpected();
```