

# Legend

optimisation  
algorithm  
kernels  
testbed  
notes  
post  
processing

**<<Interface>>**  
**ITestbenchBase**

#pde\_string: string  
#exec\_time: float  
#mem\_consumption: int  
  
+exact(x): float  
+approx(x): float  
+normL2(): float  
+solve(): None

**<<Interface>>**  
**IOptAlgoBase**

#init\_guess: np.array  
#max\_fe: int  
#min\_err: float  
  
+opt(function): (list, list, list, list)

**<<Interface>>**  
**IKernelBase**

#kernel\_type: string  
#kernel\_size: int  
  
+solution(kernels, x): float  
+solution\_x0(kernels, x): float  
+solution\_x1(kernels, x): float  
+soluiton\_x0\_x0(kernels, x): float  
+solution\_x0\_x1(kernels, x): float  
+solution\_x1\_x1(kernels, x): float

**OptAlgo{NAME}**

- \_\_init\_\_(init\_guess, max\_fe, min\_err)  
+opt(function): (list, list, list, list)

**KernelGauss**

- \_\_init\_\_()

**KernelGsin**

- \_\_init\_\_()

**FemPdeBase**

+show\_gui: bool  
#\_ngs\_ex: ngs.fem.CoefficientFunction  
#\_fes: ngs.comp.H1  
#\_gfu: ngs.comp.GridFunction  
#\_a: ngs.comp.BilinearForm  
#\_f: ngs.comp.LinearForm  
#\_g: ngs.fem.CoefficientFunction  
#\_space\_flux: ngs.comp.HDiv  
#\_gf\_flux: ngs.comp.GridFunction  
#\_mesh: ngs.comp.Mesh  
  
- \_\_init\_\_(show\_gui)  
#\_solveStep(): None  
#\_estimateError(): None

**CiPdeBase**

+pop\_histroy: list  
+fit\_history: list  
+cr\_history: list  
+f\_history: list  
#\_nc: list  
#\_nb: list  
#\_xi: list  
#\_kappa: float  
#\_weight\_reference: float  
#\_phi: list  
#\_sol\_kernel: np.array  
#\_lx: np.array  
#\_ux: np.array  
  
- \_\_init\_\_(opt\_alog, kernel, nb, nc)  
+fitness\_func(kernel): float  
+nc\_weight(xi): float  
#\_ly(float): float  
#\_uy(float): float

**FemPde0**

+max\_ndof: int  
  
- \_\_init\_\_(show\_gui, max\_ndof)  
^+solve(): None

**CiPde0**

+ max\_gen: int  
  
- \_\_init\_\_(opt\_alog, kernel, nb, nc)  
^+ fitness\_func(kernel): float  
^+ exact(x): float

**Note:**  
set of classes where  
each class represents a  
PDE 0...N of the testbed  
  
for every run in the  
experiments a new object  
is created

## Note:

different implementations of  
optimisation algorithms that  
are abstracted to a level so  
they only use 4 parameters:  
initial guess, maximum  
number of iterations,  
minimum error to terminate  
and the function to optimise

the name of the class is  
distinct where {NAME} is  
substituted

**<<Module>>**  
**PostProc**

drawGaussKernel(parameter, ggb): bool  
drawGsinKernel(parameter, ggb): bool  
saveExpObject(obj, filename): bool  
loadExpObject(filename): dict  
loadExpObjectFast(filename): dict  
plotApprox3D(kernel, parameter,  
ID, uD, name=None): None  
calcRMSE(solve\_dict): float  
statsWilcoxon(a, b, alpha=0.05): string  
plotFEDynamic(FEDynamic,  
name=None): None  
plotABSError3D(kernel, parameter,  
pdeName, ID, uD,  
name=None): None  
plotError3D(kernel, parameter,  
pdeName, ID, uD,  
name=None): None  
plotKernelAdaption(obj\_dict, title,  
name=None): None  
calcSingleERT(rundata, target): (float, float)