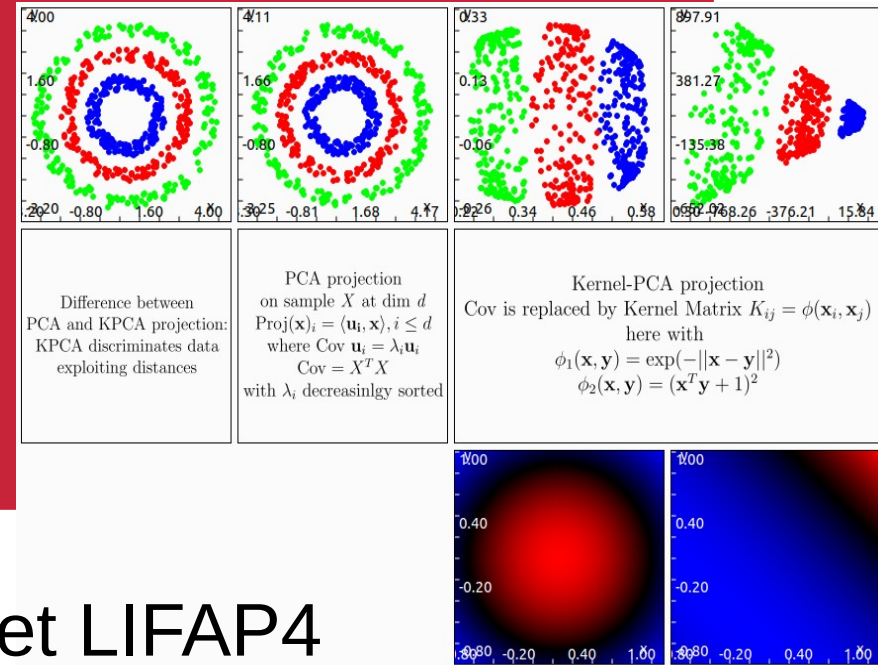


Curves & Clouds



Axel BESSY
Baptiste GENEST

Projet LIFAP4
Librairie de calcul scientifique
et visualisation de données

Diagramme de classe : Plot Window

Fonctionnement affichage :

- Plot_window
 - Plot_tab
 - Plot_frame
 - Plot_layer
 - Plottable

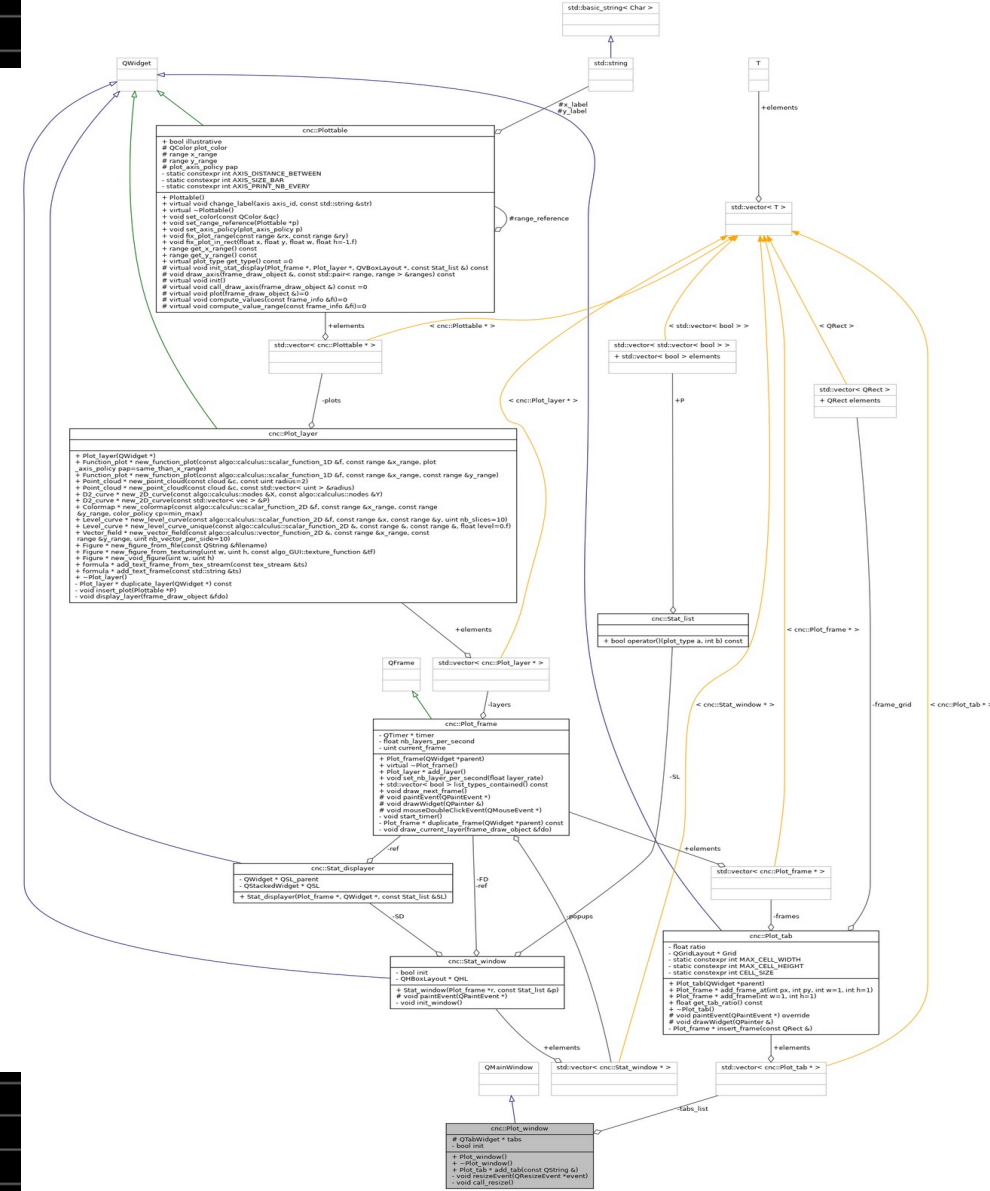
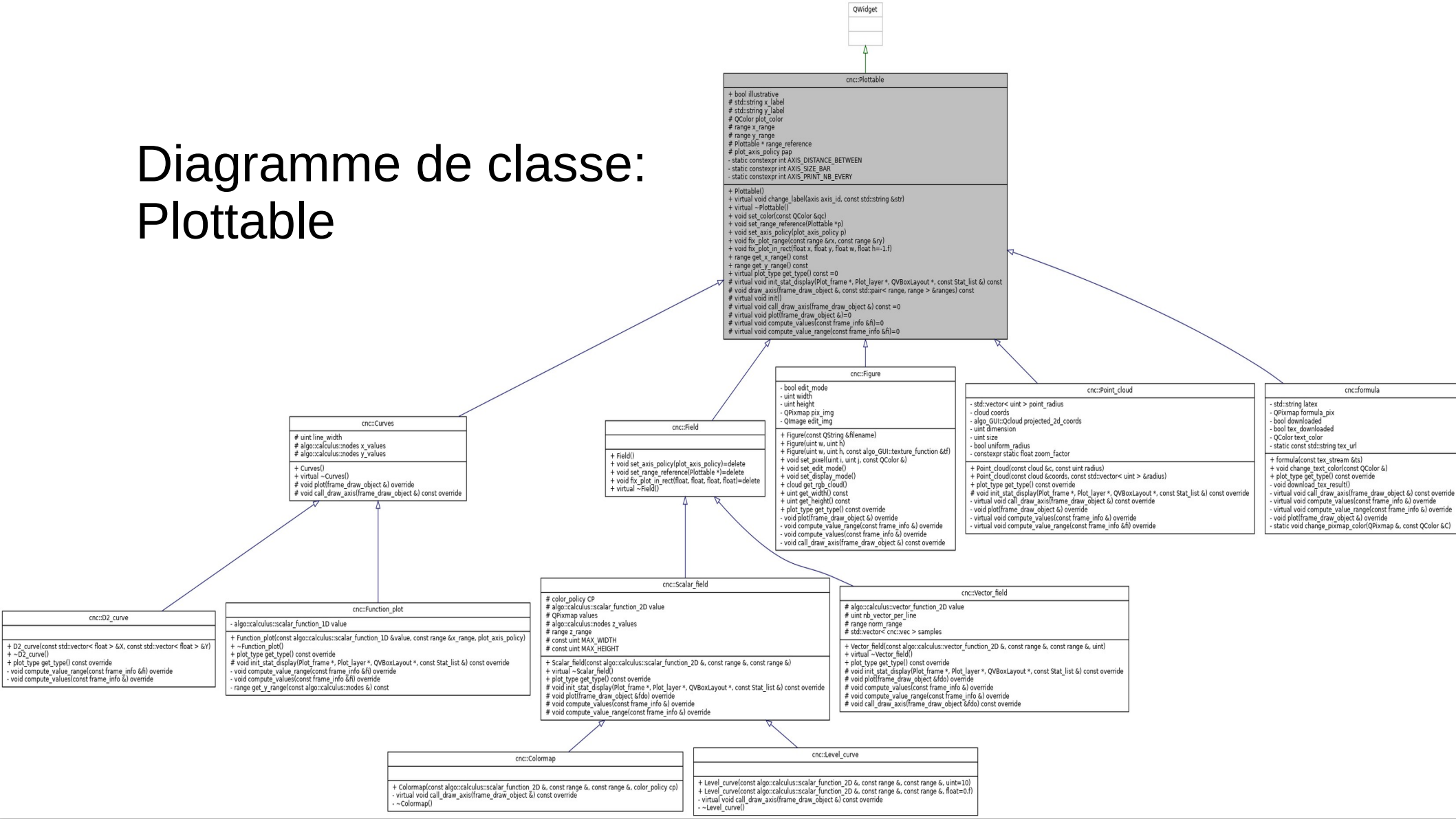


Diagramme de classe: Plottable



Focus sur la classe : Plottable

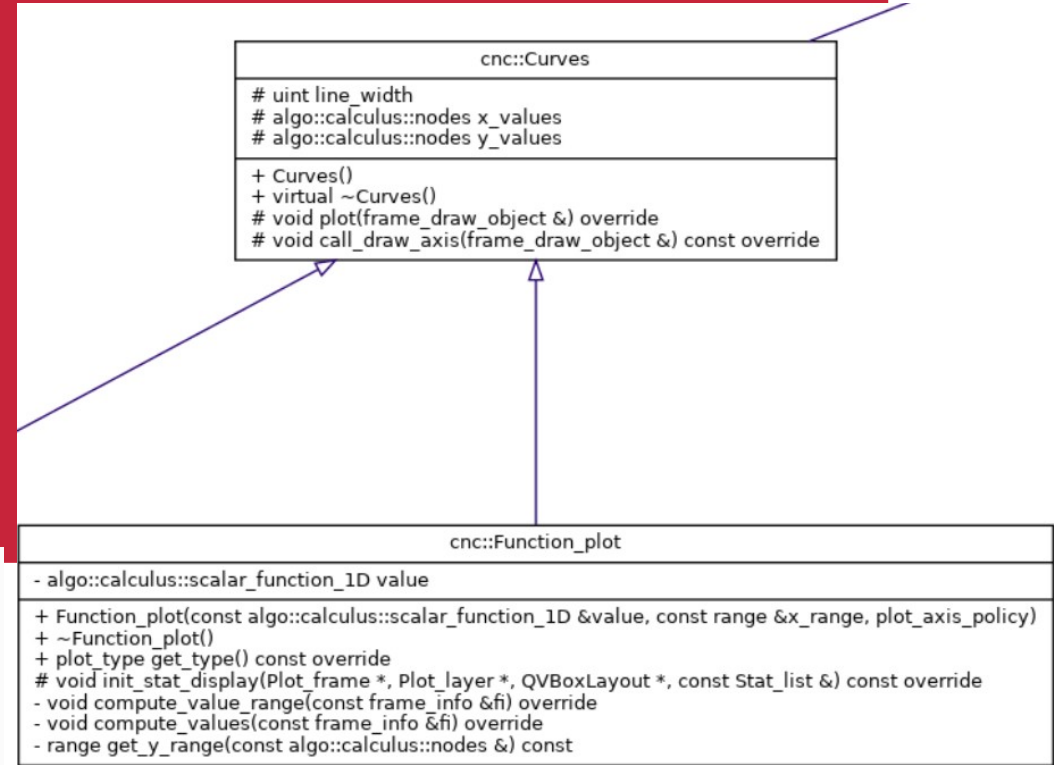
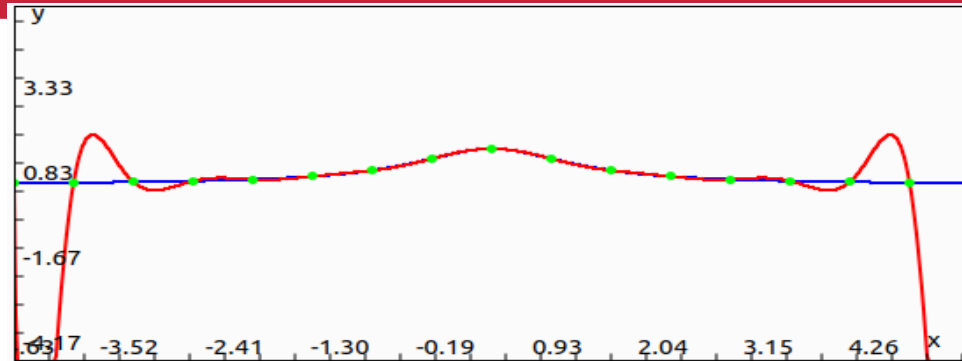
Choix de conception :
omniprésence du polymorphisme

```
cnc::Plottable

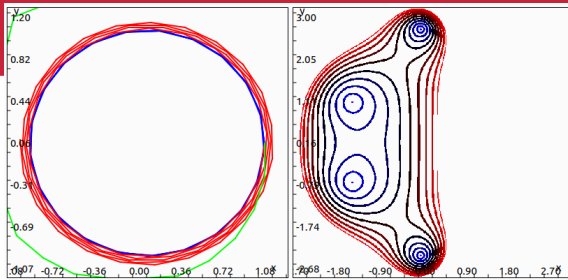
+ bool illustrative
# std::string x_label
# std::string y_label
# QColor plot_color
# range x_range
# range y_range
# Plottable * range_reference
# plot_axis_policy pap
- static constexpr int AXIS_DISTANCE_BETWEEN
- static constexpr int AXIS_SIZE_BAR
- static constexpr int AXIS_PRINT_NB_EVERY

+ Plottable()
+ virtual void change_label(axis axis_id, const std::string &str)
+ virtual ~Plottable()
+ void set_color(const QColor &qc)
+ void set_range_reference(Plottable *p)
+ void set_axis_policy(plot_axis_policy p)
+ void fix_plot_range(const range &rx, const range &ry)
+ void fix_plot_in_rect(float x, float y, float w, float h=-1.f)
+ range get_x_range() const
+ range get_y_range() const
+ virtual plot_type get_type() const = 0
# virtual void init_stat_display(Plot_frame *, Plot_layer *, QVBoxLayout *, const Stat_list &) const
# void draw_axis(frame_draw_object &, const std::pair< range, range > &ranges) const
# virtual void init()
# virtual void call_draw_axis(frame_draw_object &) const = 0
# virtual void plot(frame_draw_object &)=0
# virtual void compute_values(const frame_info &fi)=0
# virtual void compute_value_range(const frame_info &fi)=0
```

Focus sur un plot : Function_plot



Focus sur un module de calcul scientifique : Résolution d'équa diff EDO et EDP



ODE solving: comparing numerical schemes

Euler Explicit : Order 1

RK2 : Order 2

RK4 : Order 4

$\Delta t = 0.3$

Typedefs

`typedef std::vector< std::pair< vec, float > > ODE_steps`

`ODE_steps` type containing all the steps computed during **ODE** solving. [More...](#)

`typedef std::function< vec(const vec &, float)> differential`

differential derivative (here f) in the associated cauchy problem $u'(t) = f(u(t), t)$ [More...](#)

`typedef std::function< vec(const vec &, float)> ODE_scheme`

`ODE_scheme` numerical scheme to solve the **ODE**, must give u_{n+1} given u_n and t .

Functions

`std::vector< vec > extract_space_steps (const ODE_steps &S)`

`extract_space_steps` remove time information from an `ODE_steps` to only keep position (usually for plotting^). [More...](#)

ODE_scheme `build_euler_explicit (const differential & f , float dt)`

`build_euler_explicit` build the `ODE_scheme` associated with the euler explicit method (order 1) [More...](#)

ODE_scheme `build_runge_kutta_2 (const differential & f , float dt)`

`build_runge_kutta_2` build the `ODE_scheme` associated with the runge kutta 2 method (order 2) [More...](#)

ODE_scheme `build_runge_kutta_4 (const differential & f , float dt)`

`build_runge_kutta_4` build the `ODE_scheme` associated with the runge kutta 4 method (order 4) [More...](#)

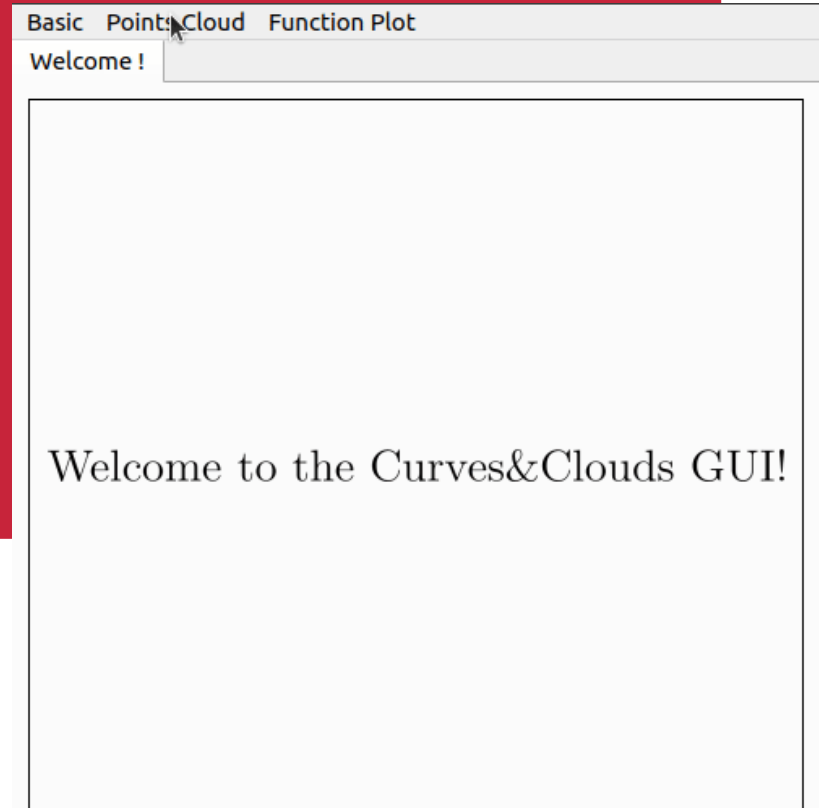
ODE_steps `solve_ODE (const ODE_scheme & s , const vec & x_0 , float t_0 , float dt , uint N)`

`solve_ODE` computes the approximation of the solution of the **ODE**, given a scheme, a starting point and time, time

Omniprésence de la programmation fonctionnelle

Focus sur une feature en WIP : C&C GUI

- Pour l'instant : parsing de fichiers csv pour affichage et analyse de nuages de points



Conclusion et avenir

- On prévoit de continuer le développement, au moins pour un usage personnel
- Cahier des charges rempli! (on avait été raisonnable...)
- Projet riche du début jusqu'à la fin pour des raisons évolutives : d'abord réflexion globale sur la structure puis réflexion locale sur l'implémentation de maths
- Ce qu'on ajoutera :
 - Choix du typage (précision des calculs)
 - Développement de l'application GUI
 - Toujours plus de maths!