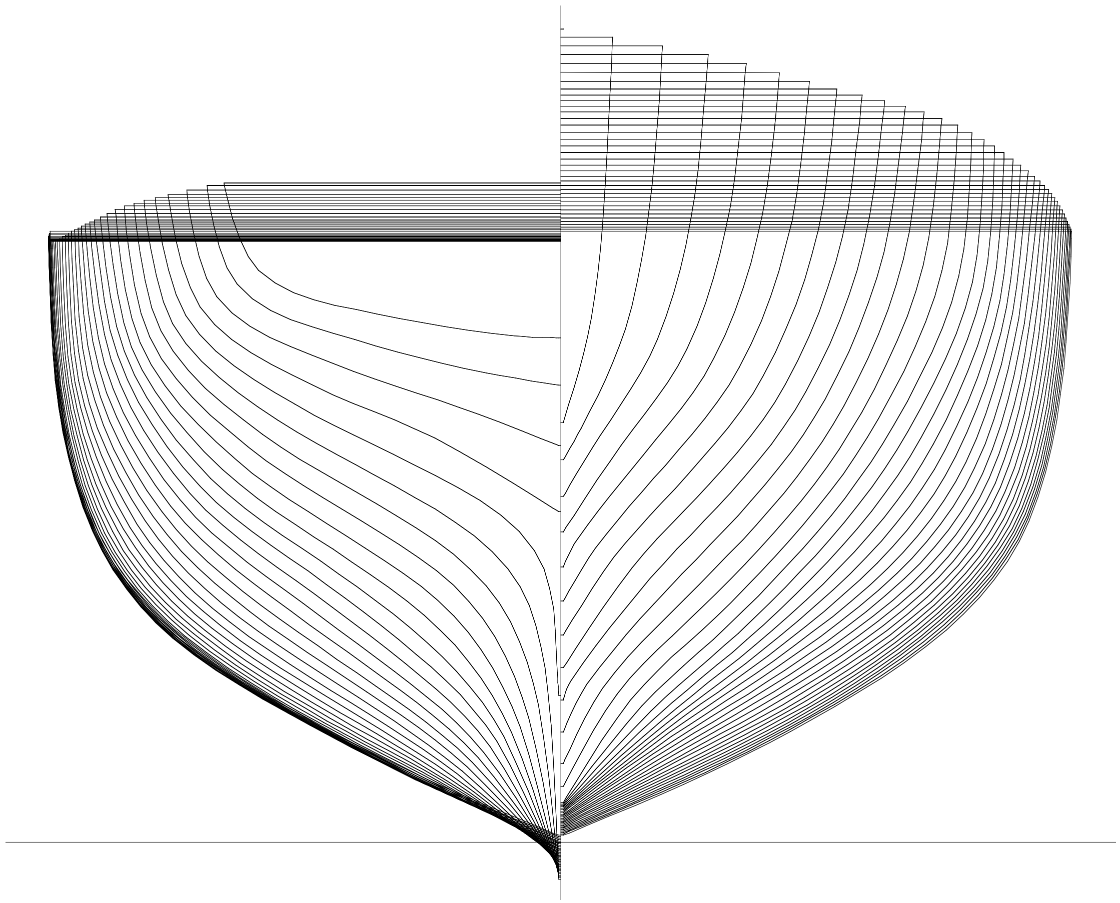
Geometric Modelling and Hydrostatics of Ships



Geometric modelling basics

Different forms of curve description

Blending functions; What are they? How do they look?

Cubic non-interpolating curves; Bezier-curves, B-splines, (H-splines)

Using non-interpolating curves as interpolators

Surface modelling by using curves (HYSS approach)

Hydrostatics - merely a question of integration

Different integration methods

Sectional integration (HYSS approach)

Longitudinal integration (HYSS approach)

Parametric curves - Algebraic form

Parametric form is almost a necessity when modelling real world geometry!
Usually one <u>curve segment</u> is described by a single normalised parameter u [0,1]

Consider a general 3D-curve with the two end points p_0 and p_1
The x,y,z-coordinates along the curve are described as polynomial functions of the single parameter u:

$$\mathbf{p}_0 = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix} \quad \mathbf{p}_1 = \begin{bmatrix} x_1 & y_1 & z_1 \end{bmatrix}$$

$$x(u) = a_{nx} \cdot u^n + \ldots + a_{1x} \cdot u + a_{0x}$$

$$x_0 = x(0) = a_{0x}$$

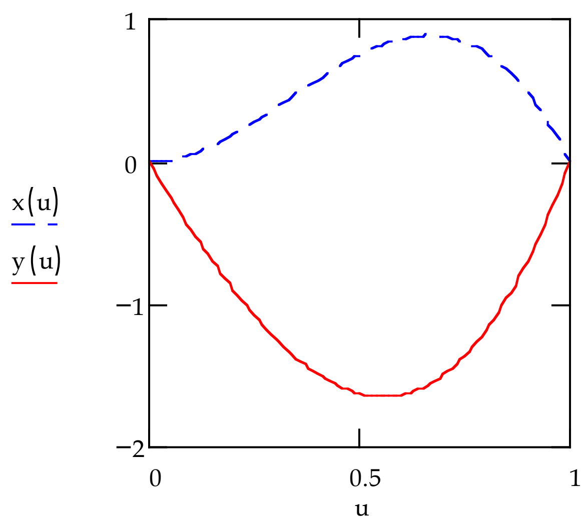
$$x_1 = x(1) = a_{nx} + \ldots + a_{1x} + a_{0x}$$

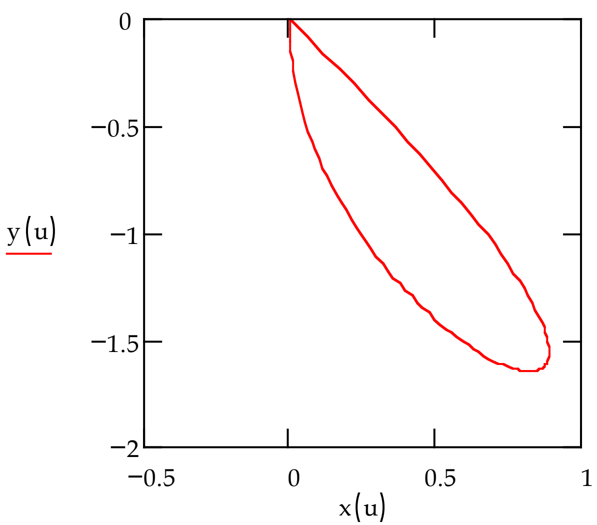
etc …

$$\mathbf{p}(u) = \mathbf{U}(u) \cdot \mathbf{A}$$

$$\mathbf{U}(u) = \begin{bmatrix} u^n & \cdots & u & 1 \end{bmatrix} \quad \mathbf{A} = \begin{bmatrix} a_{nx} & a_{ny} & a_{nz} \\ \vdots & \vdots & \vdots \\ a_{1x} & a_{1y} & a_{1z} \\ a_{0x} & a_{0y} & a_{0z} \end{bmatrix} \quad \mathbf{A} = \begin{bmatrix} \mathbf{a}_n \\ \vdots \\ \mathbf{a}_1 \\ \mathbf{a}_0 \end{bmatrix}$$

Example: $x(u) := -6 \cdot u^3 + 6 \cdot u^2 + 0 \cdot u + 0$ $\quad y(u) := 3 \cdot u^3 + 2 \cdot u^2 - 5 \cdot u + 0$





Parametric curves - Geometric Point form

The algebraic coefficients may be expressed in terms of known coordinate points along the curve. For a linear representation we need two points (the end points), for a quadratic representation we need three points, for a cubic representation four points etc. (i.e. one point more than the degree of the polynom).

$$\mathbf{p}(u) = \mathbf{U}(u) \cdot \mathbf{A}$$

point form:

$$\mathbf{P}_p = \mathbf{U}_p \cdot \mathbf{A} \quad \mathbf{P}_p = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \\ \vdots \\ \mathbf{p}_{n+1} \end{bmatrix} \quad \mathbf{U}_p = \begin{bmatrix} \mathbf{U}_1 \\ \mathbf{U}_2 \\ \vdots \\ \mathbf{U}_{n+1} \end{bmatrix} = \begin{bmatrix} u_1^{\ n} & \cdots & u_1 & 1 \\ u_2^{\ n} & \cdots & u_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ u_{n+1}^{\ \ n} & \cdots & u_{n+1} & 1 \end{bmatrix}$$

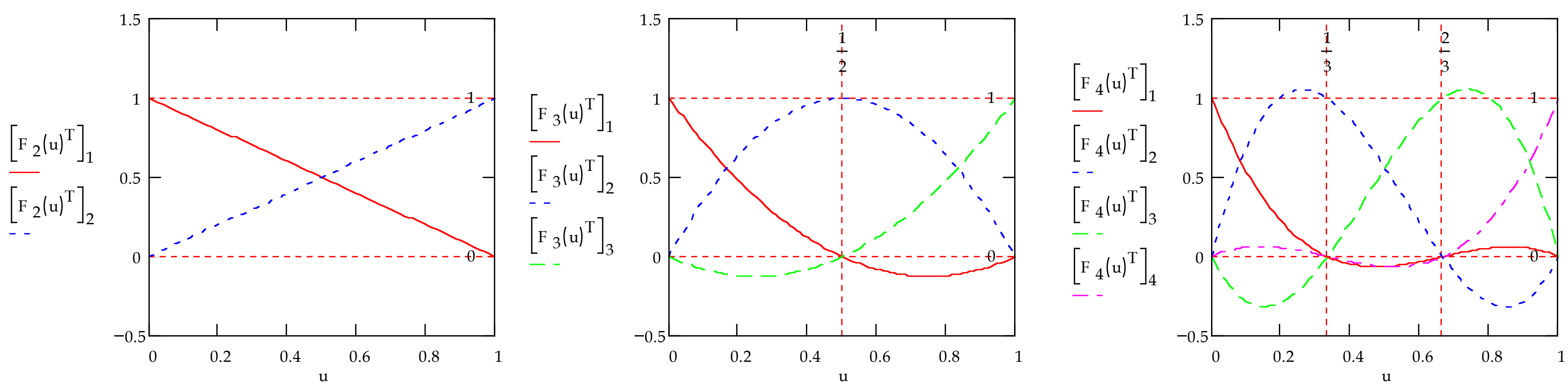
$$\mathbf{A} = \mathbf{U}_p^{-1} \cdot \mathbf{P}_p = \mathbf{N} \cdot \mathbf{P}_p$$

$$\mathbf{p}(u) = \mathbf{U}(u) \cdot \mathbf{A} = \mathbf{U}(u) \cdot \mathbf{N} \cdot \mathbf{P}_p = \mathbf{F}(u) \cdot \mathbf{P}_p$$

F(u) is a blending function (form function) which describes the relative influence from each point at a certain u value.

The sum of all components **F**(u) equals unity everywhere.

Examples of blending functions for points at equally distributed u values:



Parametric Geometric form with tangents

The geometric point form uses known point data to describe the curve (instead of explicit algebraic coefficients). However, instead of just using coordinates as in the point form, the geometric form also allows the use of derivatives. The most used geometric form is cubic because we may then use the four known data: two end-points coordinates and two derivatives to describe the full curve. By controlling derivatives at end-points we can assure that a compound curve built up from several curve segments is continuous at least up to the second derivative (smooth tangent).

$$\mathbf{p}(u) = \mathbf{U}(u) \cdot \mathbf{A}$$

$$\mathbf{p}'(u) = \mathbf{U}'(u) \cdot \mathbf{A}$$

geometric form for cubic curves:

$$\mathbf{P}_G = \mathbf{U}_G \cdot \mathbf{A} \quad \mathbf{P}_G = \begin{bmatrix} \mathbf{p}'_0 \\ \mathbf{p}_0 \\ \mathbf{p}_1 \\ \mathbf{p}'_1 \end{bmatrix} \quad \mathbf{U}_G = \begin{bmatrix} \mathbf{U}'_0 \\ \mathbf{U}_0 \\ \mathbf{U}_1 \\ \mathbf{U}'_1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix}$$

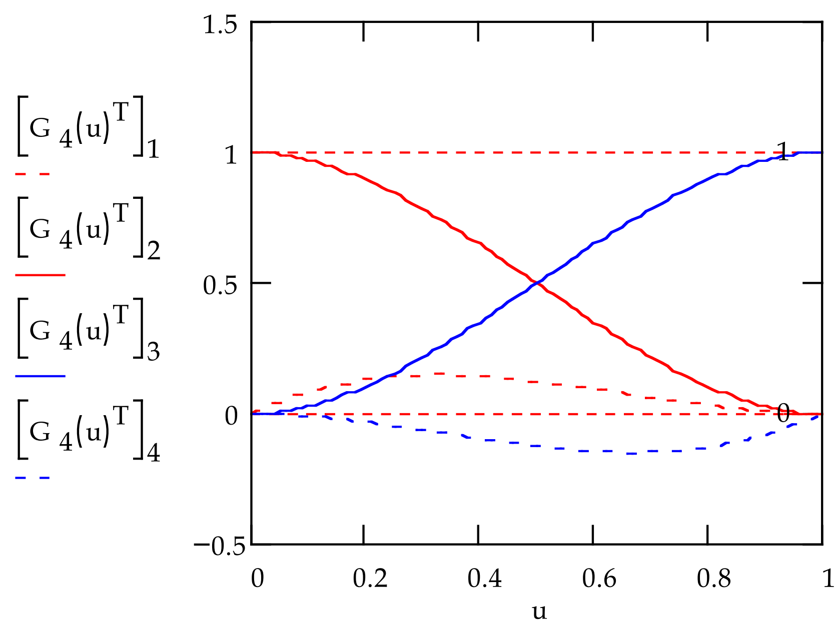
$$\mathbf{A} = \mathbf{U}_G^{-1} \cdot \mathbf{P}_G = \mathbf{M} \cdot \mathbf{P}_G$$

$$\mathbf{p}(u) = \mathbf{U}(u) \cdot \mathbf{A} = \mathbf{U}(u) \cdot \mathbf{M} \cdot \mathbf{P}_G = \mathbf{G}(u) \cdot \mathbf{P}_G$$

G(u) is a geometric blending function describing the influence from end-point data (coordinates and derivatives) along the curve.

(Note, the matrix is somewhat reordered compared to standard notation)

$$G_4(u) := \left[u^3 - 2 \cdot u^2 + u \quad 2 \cdot u^3 - 3 \cdot u^2 + 1 \quad -2 \cdot u^3 + 3 \cdot u^2 \quad u^3 - u^2 \right]$$



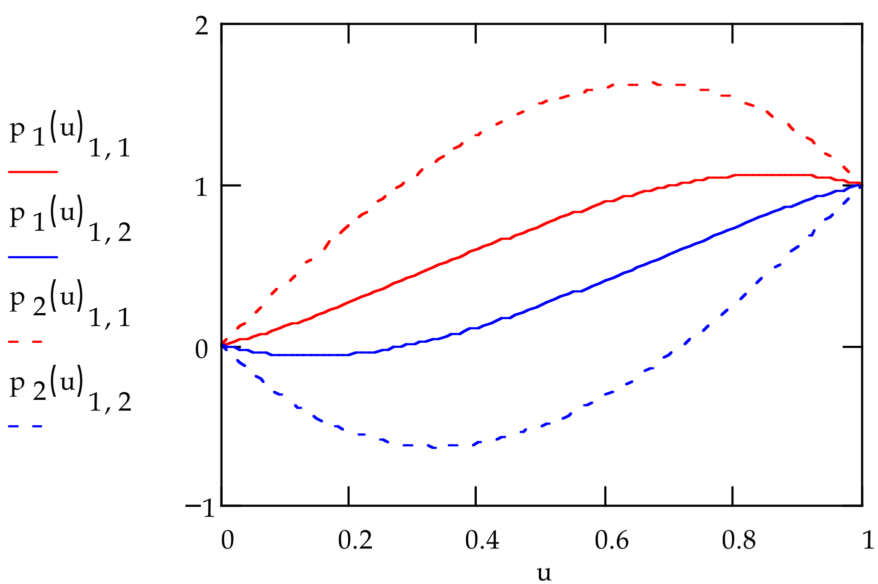
Geometric form - Example

The example compares the geometry of two curves in the x-y plane, P_{G1} and P_{G2}, both with the same coordinates (0,0) (1,1) and derivatives dy/dx = -1 at ends. However, P_{G2} has four times larger parametric derivatives dx/du, dy/du than P_{G1}.

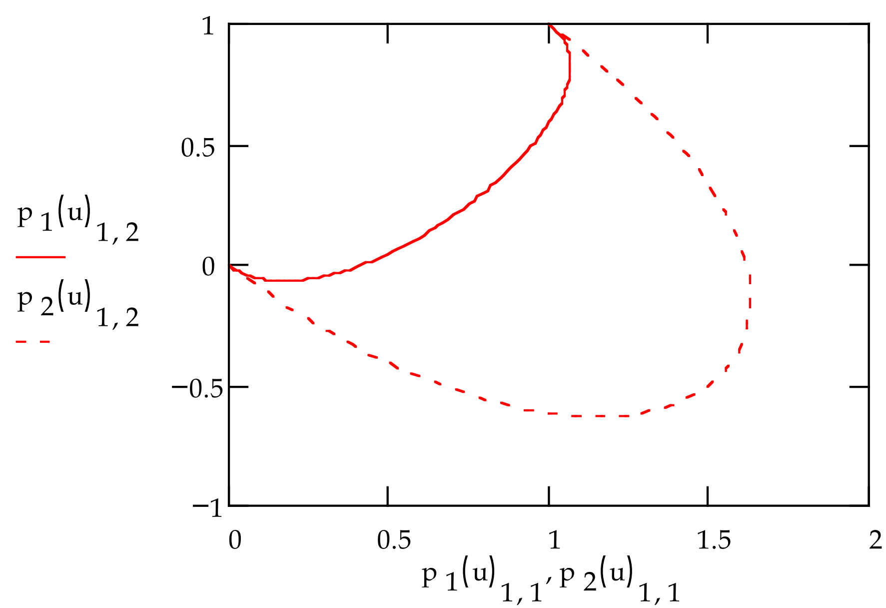
The real space plot shows that the geometric form allows control over both tangent direction and curvature (tangent length) !

$$P_{G1} := \begin{bmatrix} 1 & -1 \\ 0 & 0 \\ 1 & 1 \\ -1 & 1 \end{bmatrix} \qquad P_{G2} := \begin{bmatrix} 4 & -4 \\ 0 & 0 \\ 1 & 1 \\ -4 & 4 \end{bmatrix}$$

$$p_1(u) := G_4(u) \cdot P_{G1} \qquad p_2(u) := G_4(u) \cdot P_{G2}$$



x(u) and y(u) (parametric space)



y(x) (real space)

Bezier curves (non interpolating)

Bezier-curves are a family of curves (of different order) guided by points, but only passing through the end points. The other points are used as curve controls. The most common Bezier curves are cubic with one extra control point at each curve end which guides the tangent and tangent length (well known from most vector drawing programs). By making the second last and last point from one curve segment colinear with the first and second first points of the next segment, continuity in second derivatives are obtained. Cubic Bezier curves have the same properties as previously described cubic curves, the control point is just an easy way to adjust the tangent at the end point (which is identical to $3\cdot(p_{0c}-p_0)$ and $3\cdot(p_1-p_{1c})$).

$$\mathbf{p}(u) = \mathbf{U}(u)\cdot\mathbf{M}_{Bez}\cdot\mathbf{P}_{Bez} = \mathbf{G}_{Bez}(u)\cdot\mathbf{P}_{Bez}$$

Cubic Bezier curves:

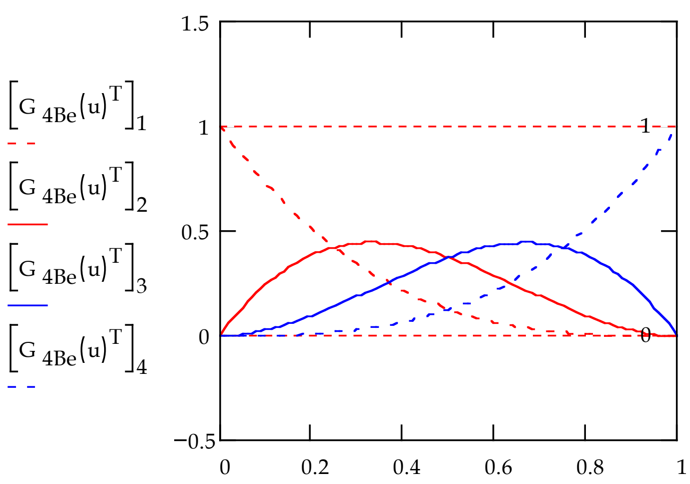
$$\mathbf{P}_{Bez} = \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{p}_{0c} \\ \mathbf{p}_{1c} \\ \mathbf{p}_1 \end{bmatrix} \qquad \mathbf{M}_{Bez} = \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

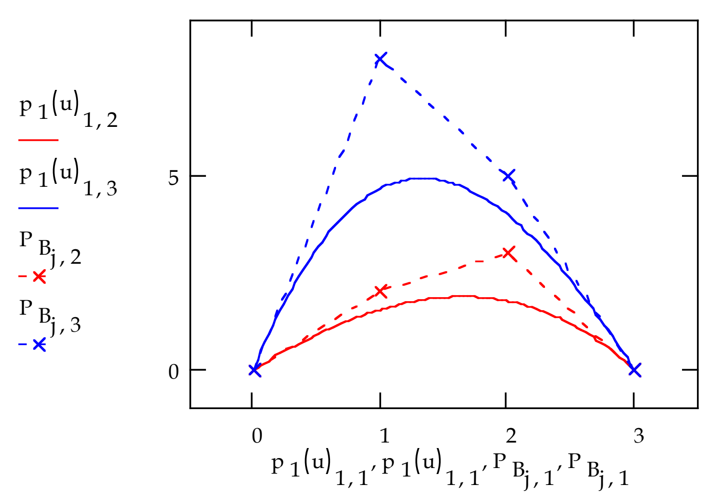
Example:

$$P_B := \begin{bmatrix} 0 & 0 & 0 \\ 1 & 2 & 8 \\ 2 & 3 & 5 \\ 3 & 0 & 0 \end{bmatrix} \qquad j := 1..4$$

$$p_1(u) := G_{4Be}(u)\cdot P_B$$

Blending functions for cubic Bezier curves:





B-splines (non-interpolating)

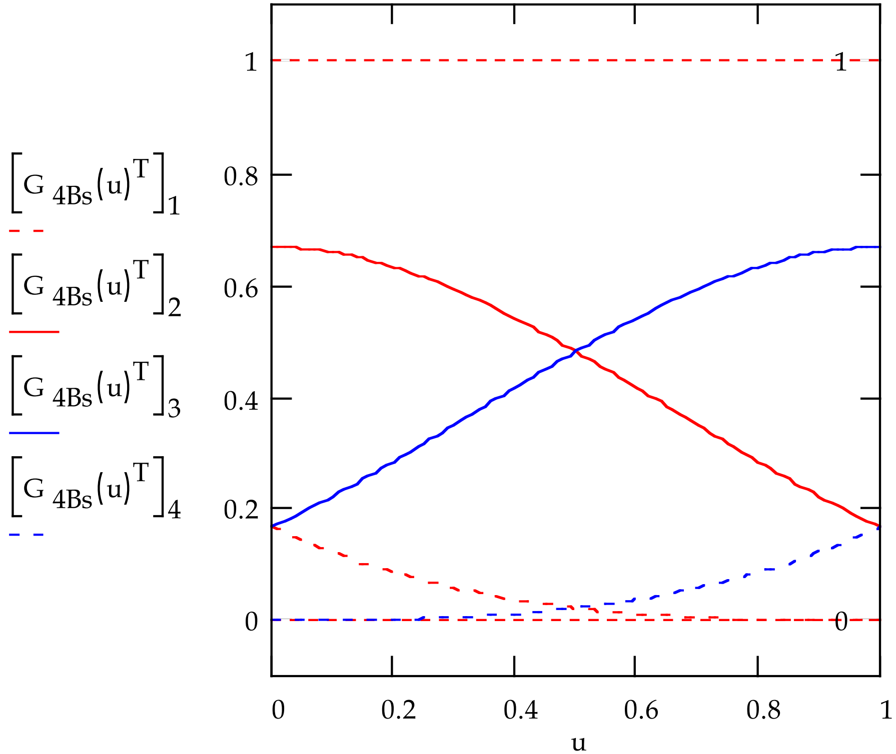
B-splines are a family of curves guided by a sequence of control points (of unlimited number) which the curve generally don't pass through. The order of the curve decides the number of control points that at the same time affects the curve. A first order curve is only affected by one control point at the time (histogram), a second order curve is affected by two points (linear variation). The most common B-splines are cubic, guided by the four closest points of the sequence.
The major advantage of the B-spline formulation is the possibility to control a complicated curve locally and still keep continuous derivatives. The major disadvantage is that you don't have control over the coordinates. However, knuckles may be introduced by letting several control points coincide.

$$\mathbf{p}(u) = \mathbf{U}(u) \cdot \mathbf{M}_{Bs} \cdot \mathbf{P}_J = \mathbf{G}_{Bs}(u) \cdot \mathbf{P}_J$$

Cubic B - splines:

$$\mathbf{P}_J = \begin{bmatrix} \mathbf{p}_{j-1} \\ \mathbf{p}_j \\ \mathbf{p}_{j+1} \\ \mathbf{p}_{j+2} \end{bmatrix} \quad \mathbf{M}_{Bs} = \frac{1}{6} \cdot \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix}$$

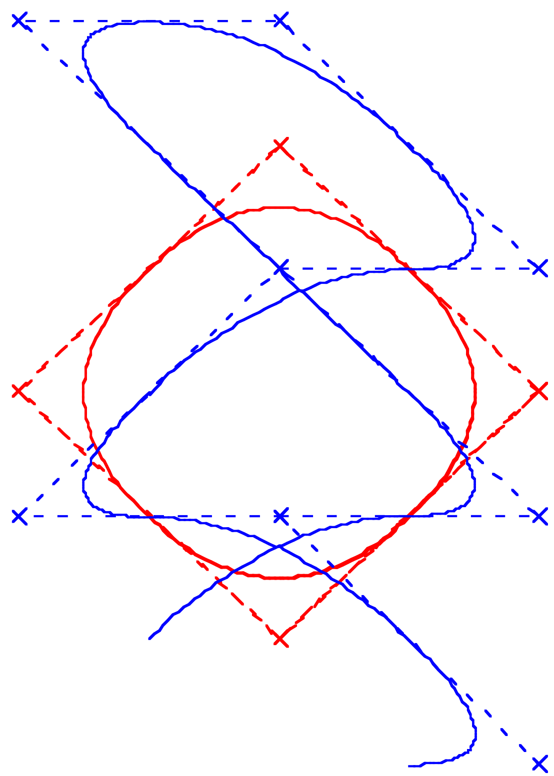
J above denotes the curve interval "between" control points p_j and p_{j+1} in a continuous curve.



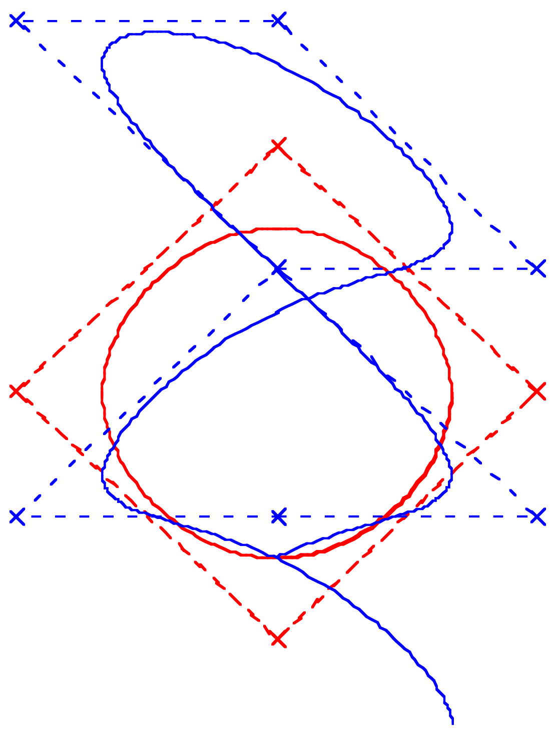
B-splines - Example

Below are two examples of B-spline curves based on a sequence of control points

Quadratic (third order):



Cubic (fourth order):



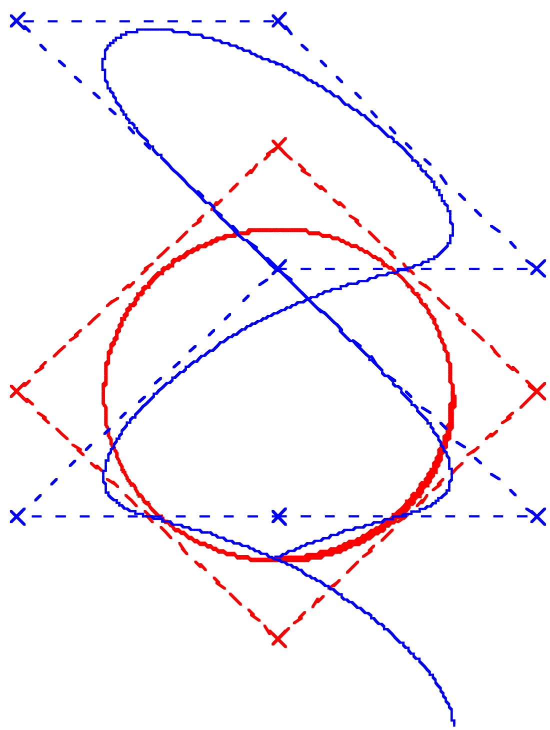
H-splines (non interpolating)

H-splines are in-house invented curves with a completely different basic formulation which is not polynomial!

(So far I haven’t found any literature about these or similar formulations so I keep it for the time beeing as a secret for possibly future patents).

The major advantage with H-splines compared to B-splines is that they are a little more smooth and have higher order continuous derivatives.

Besides this they can be handled exactly as cubic B-splines! It is therefore possible to use the same computer algorithms for both kind of splines, it is just the blending functions that are switched.



Using non-interpolating curves as interpolators

With simple matrix algebra we may transform B-spline curves to interpolating geometric curves and v.v. It is thus possible to interpolate a large sequence of points without increasing the degree of the polynom and still let the curve at any point be dependent on all the other points. Subsequently the curve may be further modified locally just by adjusting the B-spline points

Cubic B-splines:

$$\mathbf{p}(u) = \mathbf{U}(u)\cdot\mathbf{M}_{Bs}\cdot\mathbf{P}_J = \mathbf{G}_{Bs}(u)\cdot\mathbf{P}_J$$

$$\mathbf{p}'(u) = \mathbf{U}'(u)\cdot\mathbf{M}_{Bs}\cdot\mathbf{P}_J = \mathbf{G}_{Bs}{}'(u)\cdot\mathbf{P}_J$$

Geometric form for cubic curves:

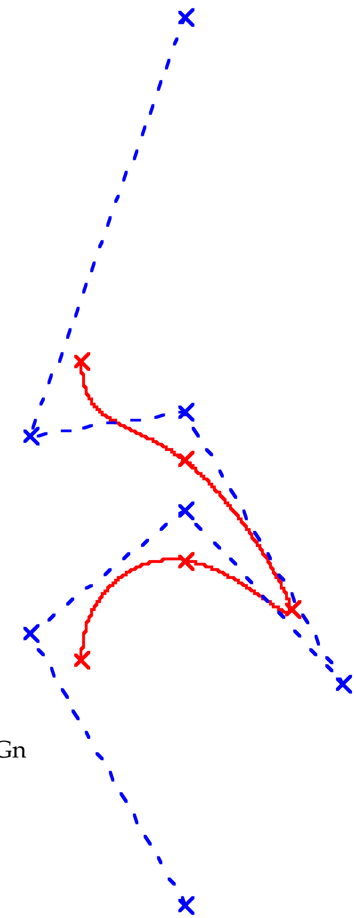
$$\mathbf{P}_G = \mathbf{U}_G\cdot\mathbf{M}_{Bs}\cdot\mathbf{P}_J = \mathbf{G}_{BsG}\cdot\mathbf{P} \quad \mathbf{P}_G = \begin{bmatrix}\mathbf{p}_0'\\ \mathbf{p}_0\\ \mathbf{p}_1\\ \mathbf{p}_1'\end{bmatrix} = \begin{bmatrix}\mathbf{G}_{Bs}{}'0\\ \mathbf{G}_{Bs0}\\ \mathbf{G}_{Bs1}\\ \mathbf{G}_{Bs}{}'1\end{bmatrix}\cdot\begin{bmatrix}\mathbf{p}_{j-1}\\ \mathbf{p}_j\\ \mathbf{p}_{j+1}\\ \mathbf{p}_{j+2}\end{bmatrix} = \frac{1}{6}\cdot\begin{bmatrix}-3 & 0 & 3 & 0\\ 1 & 4 & 1 & 0\\ 0 & 1 & 4 & 1\\ 0 & -3 & 0 & 3\end{bmatrix}\cdot\begin{bmatrix}\mathbf{p}_{j-1}\\ \mathbf{p}_j\\ \mathbf{p}_{j+1}\\ \mathbf{p}_{j+2}\end{bmatrix}$$

B-spline points that satisfy the geometric form:

$$\mathbf{P}_J = \mathbf{G}_{BsG}{}^{-1}\cdot\mathbf{P}_G$$

Generally for any sequence of points on a curve including the end derivatives we can find the B-spline points that satisfy the geometry:

$$\mathbf{P}_{Gn} = \mathbf{G}_{BsGn}\cdot\mathbf{P}_{Bs(n+2)} = \begin{bmatrix}\mathbf{p}_1'\\ \mathbf{p}_1\\ \mathbf{p}_2\\ \vdots\\ \mathbf{p}_{n-2}\\ \mathbf{p}_{n-1}\\ \mathbf{p}_n\\ \mathbf{p}_n'\end{bmatrix} = \frac{1}{6}\cdot\begin{bmatrix}-3 & 0 & 3 & \cdots & 0 & 0 & 0 & 0\\ 1 & 4 & 1 & \cdots & 0 & 0 & 0 & 0\\ 0 & 1 & 4 & \cdots & 0 & 0 & 0 & 0\\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & 0 & \cdots & 4 & 1 & 0 & 0\\ 0 & 0 & 0 & \cdots & 1 & 4 & 1 & 0\\ 0 & 0 & 0 & \cdots & 0 & 1 & 4 & 1\\ 0 & 0 & 0 & \cdots & 0 & -3 & 0 & 3\end{bmatrix}\cdot\begin{bmatrix}\mathbf{p}_{Bs1}\\ \mathbf{p}_{Bs2}\\ \mathbf{p}_{Bs3}\\ \vdots\\ \mathbf{p}_{Bs(n-1)}\\ \mathbf{p}_{Bsn}\\ \mathbf{p}_{Bs(n+1)}\\ \mathbf{p}_{Bs(n+2)}\end{bmatrix} \Rightarrow \mathbf{P}_{Bs(n+2)} = \mathbf{G}_{BsGn}{}^{-1}\cdot\mathbf{P}_{Gn}$$

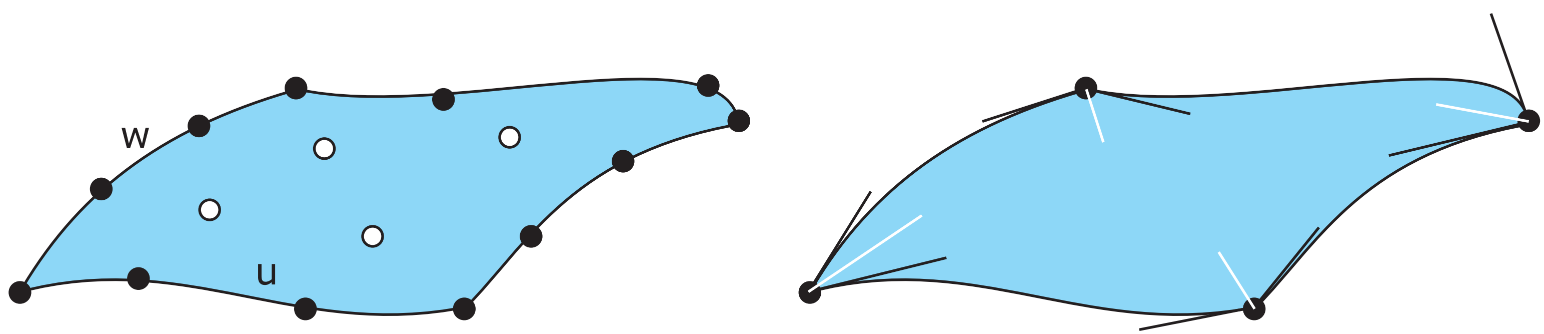


Parametric surface modelling

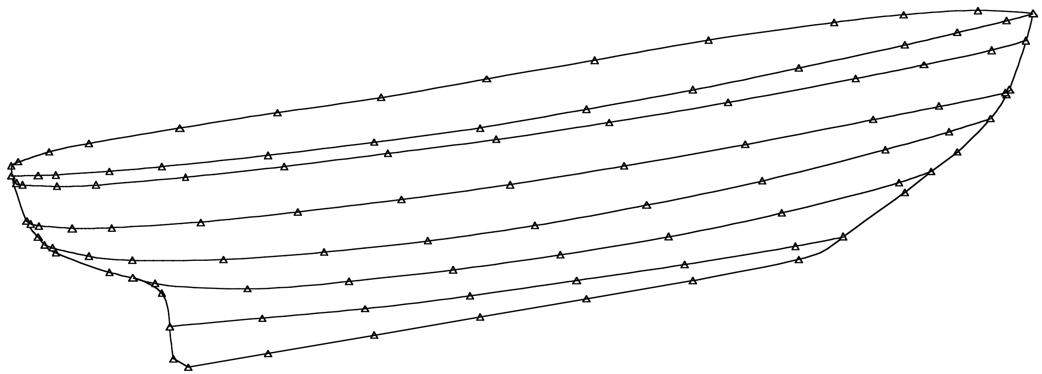
Surface modelling may in principal be performed by the same methods as curve modelling. The coordinates of a surface is described by two parameters, **p**(u,w), instead of just one. The geometric equations for a cubic surface may be solved from 16 or 12 points on the surface (similar to the point form for curves) or may be solved from 4 corner coordinates together with 2 or 3 derivatives (similar to the previously described general geometric form for curves). The four "internal" points or the four "extra" derivatives are not necessary but allows for a more detailed description of the geometry "within" the surface.

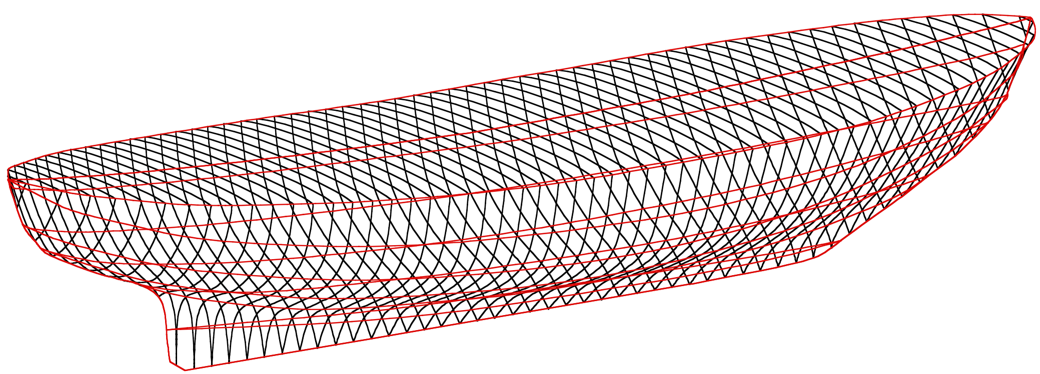
Cubic B-splines or similar formulations are commonly used also for surface modelling.

If one of the surface parameters is fixed to a space coordinate direction, surface modelling may also be directly performed by curves only by adding information of the tangent in the "fixed" direction along the curve. This approach has several advantages when dealing with ship hulls.



Surface modelling from Lines (HYSS approach)





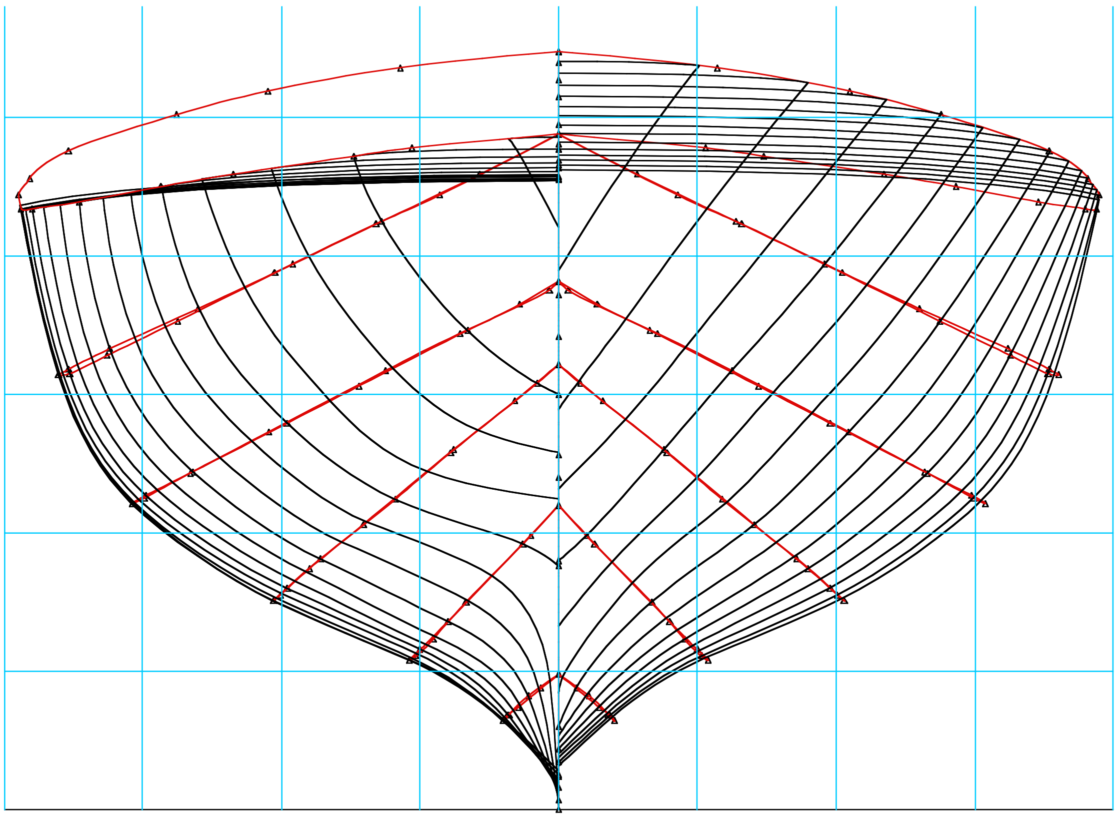
1 A reasonable number of longitudinal **Lines** are defined with points giving x,y,z-coordinates. In this case 8 lines: the Bottom CL, 5 Diagonals, Sheer Line and Deck CL. The curvature of the lines is here automatically calculated but may also be modified (tangent length and direction). Lines are H- or B-splines or straight.

2 For each Line, the tangent and direction in the sectional plane is automatically calculated from the position of the surrounding lines or may be specified (varying as a polynomial of third degree).

3 The lines are (after check) mirrored in the CL-plane.

4 **Sections** are cut from the lines by picking coordinates and tangent data from the lines in their defined order. In the example, the tangent length of Sheer Line and Keel is set to zero (knuckle) and the tangent length of Deck CL is increased to create a camber.

Surface Modelling from Lines; pros and cons

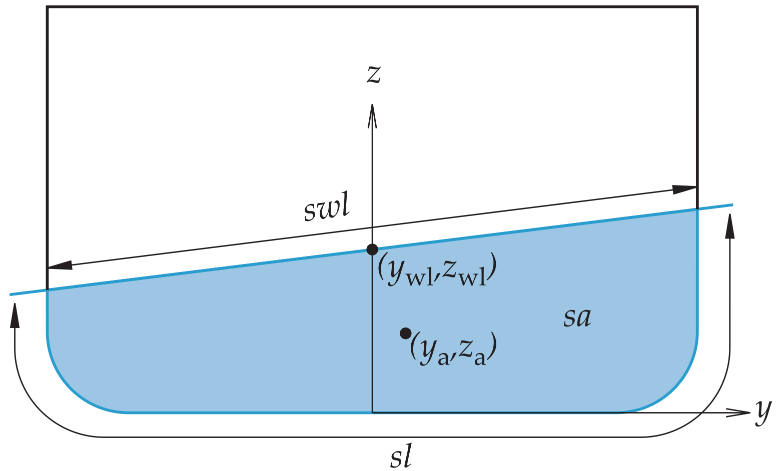


Pros:
- Limited, efficient input
- Full control over coordinates
- Lines define the natural boundary of a hull
- Lines are smooth
- Full control over sections which are used for calculation of hydrostatics
- Easy to model knuckles and transitions between knuckles and curved body
- Multihull sections no problem
- Lines may be created (cut) from sections, and thus it is possible to create a full surface model from an offset table and a stem-stern contour.

Limitations:
- Lines must be defined with increasing longitudinal coordinate where sections are cut
- Lines must be defined in the correct sequence along the sections

Cons:
- No automatic coordinate release (smoothing)
- Discontinuities may arise when Lines ends on a surface

Hydrostatics - merely a question of integration



$$\nabla = \int_{L_{WL}} sa \, dx$$

$$LCB = \frac{1}{\nabla} \int_{L_{WL}} sa \, x \, dx$$

$$WA = \int_{L_{WL}} swl \, dx \qquad LCF = \frac{1}{WA} \int_{L_{WL}} swl \, x \, dx$$

$$TCB = \frac{1}{\nabla} \int_{L_{WL}} sa \, y_a \, dx$$

$$BM_T = \frac{I_{WAx}}{\nabla} = \frac{1}{\nabla} \int_{L_{WL}} \frac{swl^3}{12} dx$$

$$KB = \frac{1}{\nabla} \int_{L_{WL}} sa \, z_a \, dx$$

$$BM_L = \frac{I_{WAy}}{\nabla} = \frac{1}{\nabla} \left(\int_{L_{WL}} swl \, x^2 \, dx - WA \, LCF^2 \right)$$

$$S = \int_{L_{WL}} sl \, dx$$

Numerical integration methods

Numerical integration is a standard procedure in engineering applications. However, for integration of ship geometries with no explicit mathematical description and with possible steps and knuckles, standard procedures are not always feasible.

Step by step (discretisation)

$$\int_{x_0}^{x_1} y\,dx$$

Polynomial approximations:

- Newton-Cotes (series of values at equal steps multiplied with specific factors, i.e. Simpsons rules)

- Tchebycheff (series of values at specific positions)

- Gauss (series of values at specific positions multiplied with specific factors)

- Direct partial polynomial identification with analytical integration

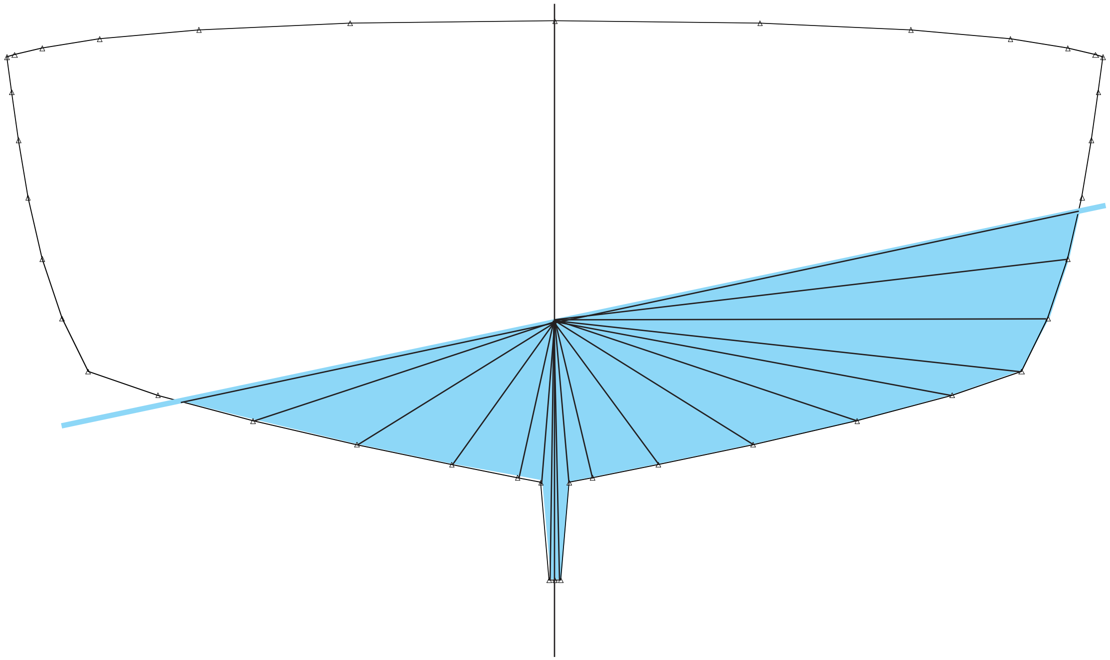
Parametric integration:

- Analytical from the geometric description

- Step by step (in parametric space)

$$\int_{x_0}^{x_1} y\,dx = \int_0^1 y(u) \cdot \frac{dx(u)}{du}\,du$$

Integration methods in HYSS



Longitudinal integration
of sectional values is performed with a partial third degree polynomial description with limitations to assure stable performance.

- Enables uneven spacing between sections

- Allowance for steps (deckhouses) by adding close sections around the step

- Precision dependent on number of sections

Sectional integration
is performed step-by-step in the parametric description around the crosspoint between water line and CL

This enables:

- Unlimited precision, dependent only on the number of steps for each curve segment

- Multihull integration automatically

- Absolutely stable performance

- No problem with a mixture of curved segments, straight segments and knuckles

- Independent of heel angle