```
> restart;
  with(LinearAlgebra) :
  solveSystem := (a, b)→LinearSolve(a, b) : with(CurveFitting) :
```

```
> # Кубический сплайн
  Cubic := proc(f)
   local n := 10 :
   local i;
   local segment := 0..1 :
   local h := 1/n ;

   local xs := seq(i, i = segment, h) ;
   local ys := seq(f(i), i = segment, h) ;

   local A_init := (i, j)→
      if i = j and i ≠ 1 and i ≠ n + 1 then 4 h
      elif (i − j = 1 or i − j = −1) and i ≠ 1 and i ≠ n + 1 then h
      elif (i = 1 and j = 1) or (i = n + 1 and j = n + 1) then 1
      else 0; end if;

   local A := Matrix(n + 1, A_init);

  local vector := 6· Vector(n + 1, j→if (j = 1 or j = n + 1) then 0 else 1/h ((ys[j + 1]
      − ys[j]) − (ys[j ]− ys[j − 1])); end if);

   local a := seq(ys[i], i = 2 ..n + 1);
   local c := solveSystem(A, vector);
   local b := seq((ys[i] − ys[i − 1])/h + c[i]·h/3 + c[i − 1]·h/6, i = 2 ..n + 1);
   local d := seq((c[i] − c[i − 1])/h, i = 2 ..n + 1);

   local S := (i, x) → a[i] + b[i]·(x − xs[i + 1]) + c[i + 1]/2 ·(x − xs[i + 1])^2 + d[i]/6 ·(x
      − xs[i + 1])^3;

   local P := proc (x)
     for i from 1 to n do
       if (xs[i] ≤ x ≤ xs[i + 1]) then return S(i, x) end if; end do;
  end proc;

  return x→ P(x) ;

  end proc:
```

```
> #B-сплайн
Bspline := proc(f)
local eps := 10^(-9) :
local n := 12 :
local segment := 0 ..1 :
local i;
local h := 1/(n - 2);

local xs := [ -2 ·eps, -eps, seq(i, i=segment, h), eps + 1, 2·eps + 1 ];

local k := j →
   if j = 1 then f(xs[j])
   elif j = n then f(xs(n + 1))
   else 1/2 · ( -f(xs[j + 1]) + 4·f( (xs[j + 1] + xs[j + 2])/2 ) - f(xs[j + 2]) ); end if;

local B0 := (i, x) → piecewise(xs[i] ≤ x < xs[i + 1], 1, 0);
local B1 := (i, x)→ (x - xs[i])/(xs[i + 1] - xs[i]) ·B0(i, x) + (xs[i + 2] - x)/(xs[i + 2] - xs[i + 1]) ·B0(i + 1, x);
local B2 := (i, x)→ (x - xs[i])/(xs[i + 2] - xs[i]) ·B1(i, x) + (xs[i + 3] - x)/(xs[i + 3] - xs[i + 1]) ·B1(i + 1, x);

local S := x → sum(k(i)·B2(i, x), i = 1 ..n);

return x → S(x);

end proc:

> deviations := proc (func1, func2)
local segment := 0 ..1 :
local n := 10 :
local h := 1/(10·n);
local maxError := 0;
local grid := seq(i, i=segment, h) ;
local node, i, errors;

for node in grid do
errors := abs(evalf(func1(node) - func2(node)));
if maxError < errors then maxError := errors; end if; end do;
return maxError;
end proc:
```

> # `*Сравнение встроенные сплайнов* с *самописными сплайнами*

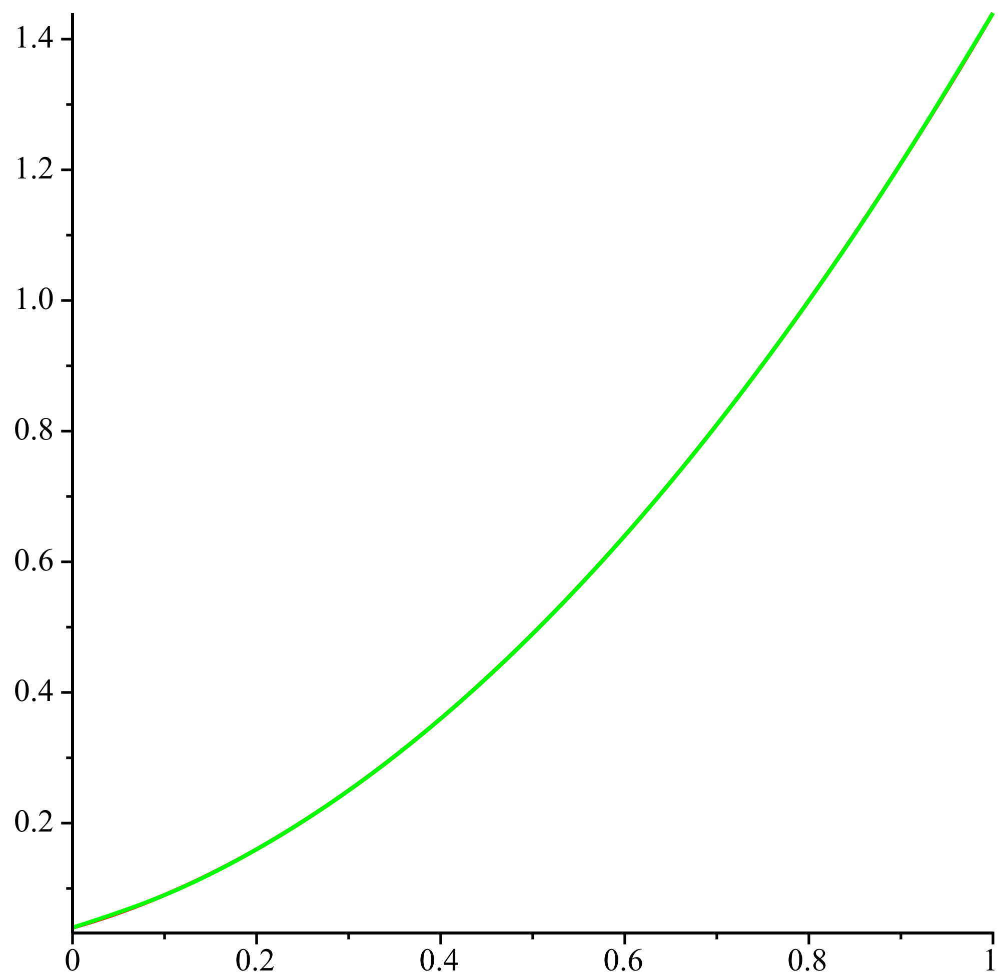
$f := x \to \left(x + \frac{1}{5}\right)^2 :$

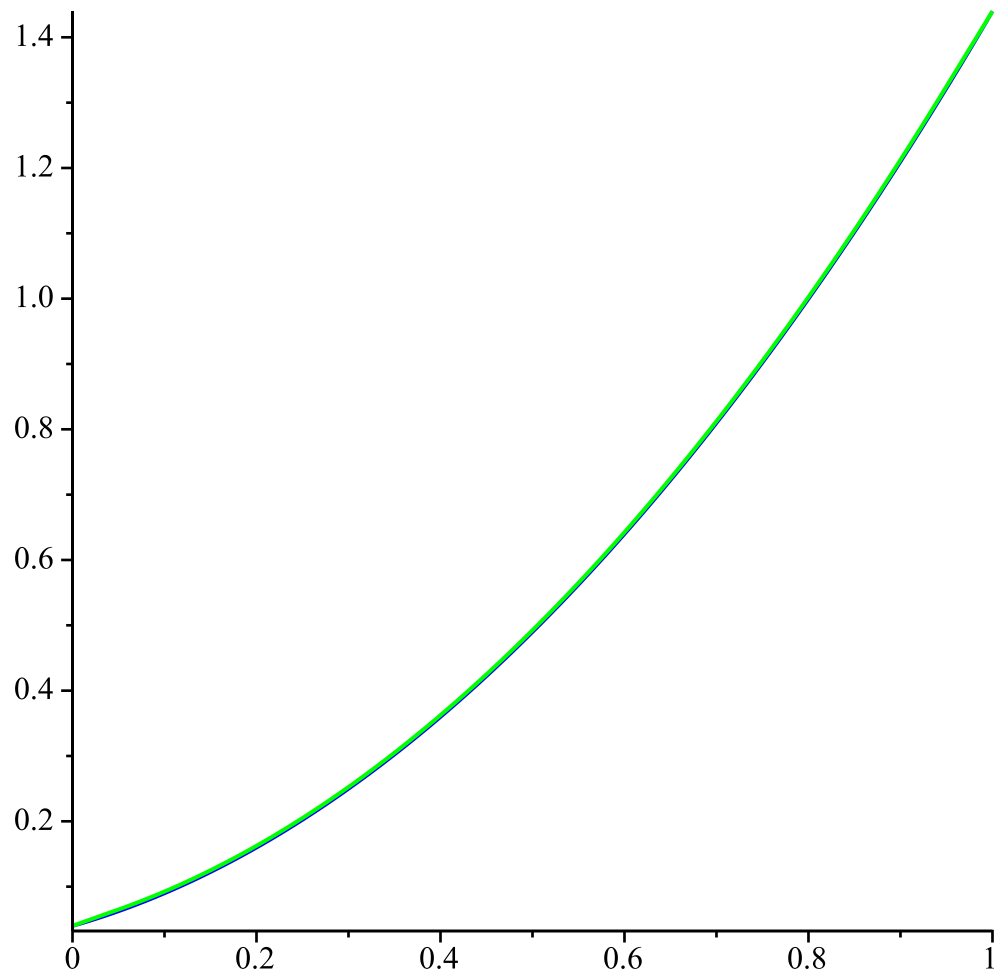
$mapleCubicSplineInterp := \mathrm{x} \to Spline\left(\left[\mathrm{seq}\left(\mathrm{i}, \mathrm{i} = 0 \,..1, \frac{1}{10}\right)\right], \left[\mathrm{seq}\left(\mathrm{f(i)}, \mathrm{i} = 0 \,..1, \frac{1}{10}\right)\right], \mathrm{x}, \mathrm{degree} = 3\right) :$

$mapleBSplineInterp := BSplineCurve\left(\left[-2\cdot 10^{-9}, -10^{-9}, seq\left(i, i = 0 \,..1, \frac{1}{10}\right), 1 + 10^{-9}, 1 + 2\cdot 10^{-9}\right], \left[f(0), f(0), seq\left(f(i), i = 0 \,..1, \frac{1}{10}\right), f(1), f(1)\right], x, order = 3\right) :$

$plot(\,[\mathrm{f},\ Cubic(f), \mathrm{mapleCubicSplineInterp}\,], 0\,..1, \mathrm{color} = [\mathrm{red, blue, green}]\ );$
$plot(\,[\mathrm{f}, Bspline(f), \mathrm{mapleBSplineInterp}\,], 0\,..1, \mathrm{color} = [\mathrm{red, blue, green}]\,);$

Warning, (in mapleCubicSplineInterp) `i` is implicitly declared local





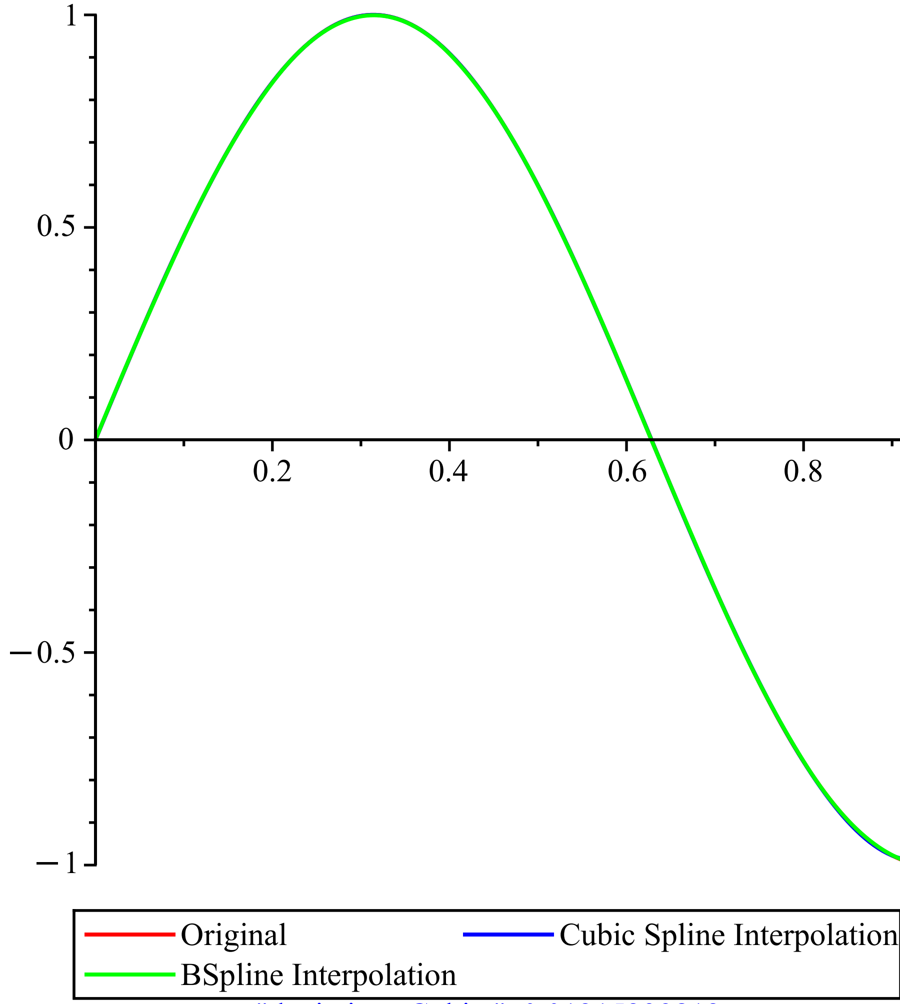
> *# Как видно из графиков, самописная <u>реализация сплайнов хороша</u>*

> **# Эксперименты**

> ***# Рассмотрим невысокочастотную функцию и убедимся, что сплайны будут хорошо интерполировать ее***

```
f := x → sin(5 x);
plot([f, Cubic(f) , Bspline(f) ], 0 ..1, color = [red, blue, green], legend = ["Original",
    "Cubic Spline Interpolation", "BSpline Interpolation"]) ;
print("deviations Cubic:", deviations(f, Cubic(f) ) ) :
print("deviations Bspline:", deviations(f, Bspline(f) ) ) :
```

$$f := x \mapsto \sin(5 \cdot x)$$



"deviations Cubic:", 0.01215239812

"deviations Bspline:", 0.001281730953 **(1)**

> ***# Рассмотрим высокочастотную функцию и убедимся , что сплайны при высокочастотной для таких функций не будут соответствовать реальности, тк коэффиценты не успевают реагировать на постоянные скачки функции***

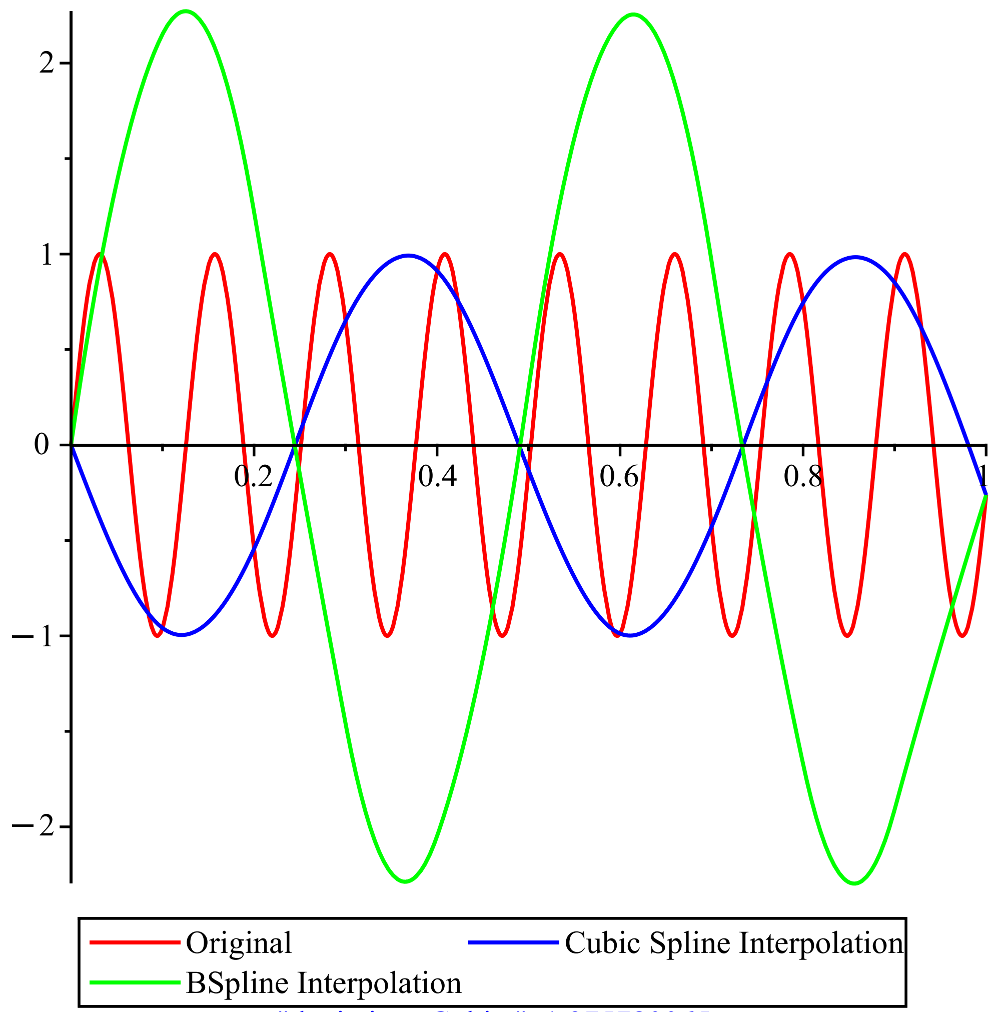
$f := x \to \sin(50\,x);$

plot([f, Cubic(f) , *Bspline*(*f*)] , 0 ..1, color = [red, blue, green], legend = ["Original", "Cubic Spline Interpolation", "BSpline Interpolation"]) ;

print("deviations Cubic:", *deviations*(*f*, Cubic(f))) :

print("deviations Bspline:", *deviations*(*f*, *Bspline*(*f*))) :

$$f := x \mapsto \sin(50 \cdot x)$$



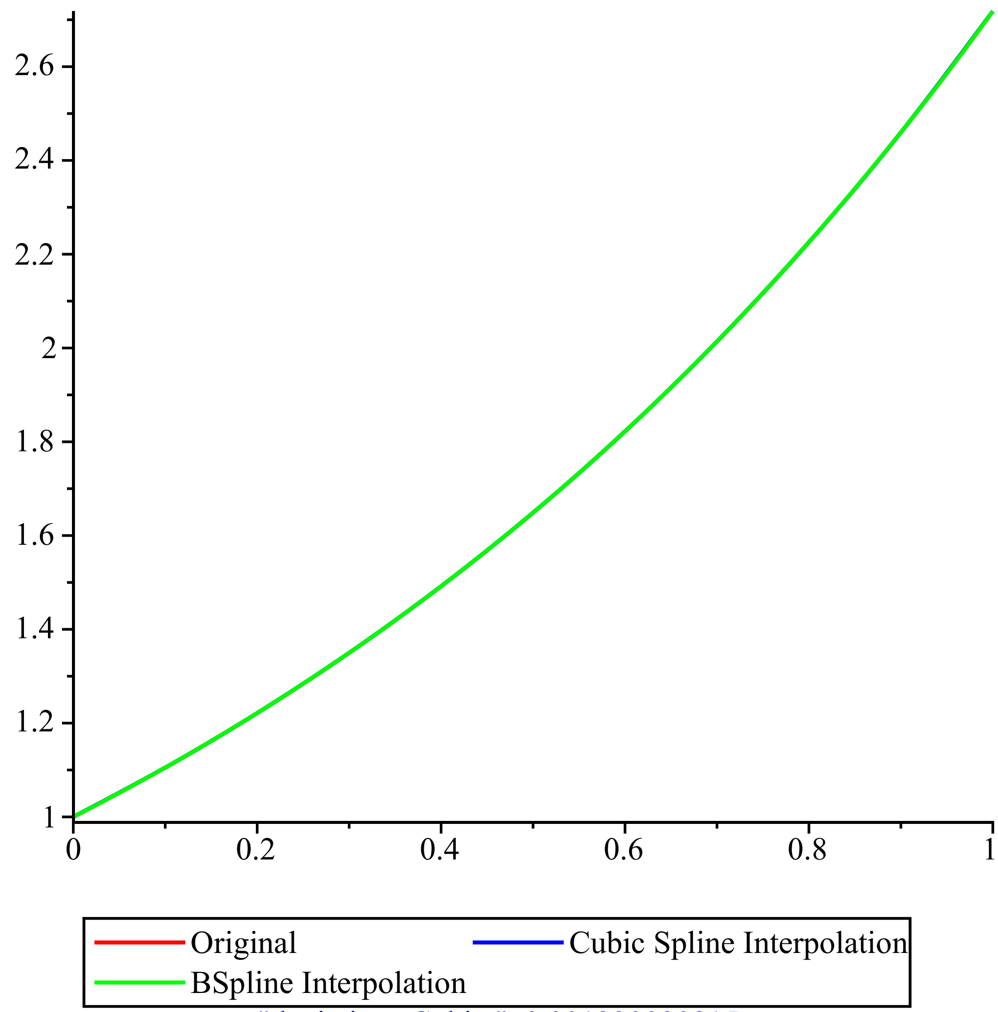
"deviations Cubic:", 1.975780065

"deviations Bspline:", 3.205291497 **(2)**

> ***#Рассмотрим функцию от экспоненты***

```
f := x →exp(x);
plot([f, Cubic(f) , Bspline(f) ], 0..1, color = [red, blue, green], legend = ["Original",
    "Cubic Spline Interpolation", "BSpline Interpolation"]) ;
print("deviations Cubic:", deviations(f, Cubic(f) ) ) :
print("deviations Bspline:", deviations(f, Bspline(f) ) ) :
```

$$f := x \mapsto e^{x}$$



"deviations Cubic:", 0.001330090815

"deviations Bspline:", 0.00002299448 **(3)**

\>

\> # ***Как видно из графиков и из ошибок оба сплайна отлично справились с функциями, котороые не имеют постоянных скачков***

\>