```
> restart;
```

```
> # ``Основные значения
with(LinearAlgebra) :
solveSystem := (a, b)→LinearSolve(a, b) :

i, j :
n := 10 :
section := 0 ..1 :
border1 := 0 :
border2 := 0 :
eps := 10^(−9) :
```

```
> # Кубические сплайны
CubicSplineInterpolation := proc(n, section, border1, border2, f)

 local i;
 local h := 1/n ;

 local grid := [seq(i, i = section, h)];
 local values := [seq(f(i), i = section, h)];

 local tridiagonalMatrixInit := (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 tridiagonalMatrix := Matrix(n + 1, tridiagonalMatrixInit);

 local rhVectorInit := i→ if i = 1 then border1 elif i = n + 1 then border2
    else 6·(values[i + 1] − 2·values[i] + values[i − 1])/h ; end if;
 local rhVector := Vector(n + 1, rhVectorInit);

 local a := seq(values[i], i = 2 ..n + 1);
 local c := solveSystem(tridiagonalMatrix, rhVector);
 local b := seq((values[i] − values[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 − grid[i + 1]) + c[i + 1]/2·(x − grid[i + 1])^2 + d[i]/6
```

```
    ·(x − grid[i + 1])^3;

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

return x→ P(x) ;

end proc:
```

> # #B-сплайны

```
BSplineInterpolation := proc(n, section, eps, f)

local i;
local h := 1/(n − 2);
   # У нас n передается в функцию с учетом "фиктивных" точек, поэтому чтобы
   вычислить шаг мы от n отнимаем 2.

local grid := [−2 ·eps, −eps, seq(i, i = section, h), eps + 1, 2·eps + 1 ];
local values := [f(0), f(0), seq(f(i), i = section, h), f(1), f(1)];

local k := j →
  if j = 1 then f(grid[j])
  elif j = n then f(grid(n + 1))
  else 1/2 · (−f(grid[j + 1]) + 4·f((grid[j + 1] + grid[j + 2])/2) − f(grid[j + 2]));
   end if; #Coefficient.

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

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

return x → S(x);

end proc:
```

```
> # Отображение результатов
ShowGraphs := proc(f, CubicSplineInterp, BSplineInterp)
 local res := plot([f, CubicSplineInterp, BSplineInterp], 0..1, color = [red, blue, green], legend
      = ["Original", "Cubic Spline Interpolation", "BSpline Interpolation"]);
 return res;
end proc:
```

```
> # Вычисление ошибки
 ComputeIterpolationError := proc (func1, func2, n, section)
 local h := 1/(10·n);
 local maxError := 0;
 local grid := [seq(i, i = section, 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:
```

```
> #``Проверка корректности построенных сплайнов
   посредством сравнения их со стандратными
   средствами Maple.
```

$f := x \to x^2 \sin\left(\frac{1}{x - 0.0001}\right)$:

```
myCubicSplineInterp := CubicSplineInterpolation(n, section, border1, border2, f) :
myBSplineInterp := BSplineInterpolation(n, section, eps, f) :
```

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

$with(CurveFitting) : mapleBSplineInterp := BSplineCurve\left(\left[-2\cdot eps, -eps, seq\left(i, i = section,\right.\right.\right.$
$\left.\left.\frac{1}{n}\right), 1 + eps, 1 + 2\cdot eps\right], \left[f(0), f(0), seq\left(f(i), i = section, \frac{1}{n}\right), f(1), f(1)\right], x, order$
$\left. = 3\right)$:

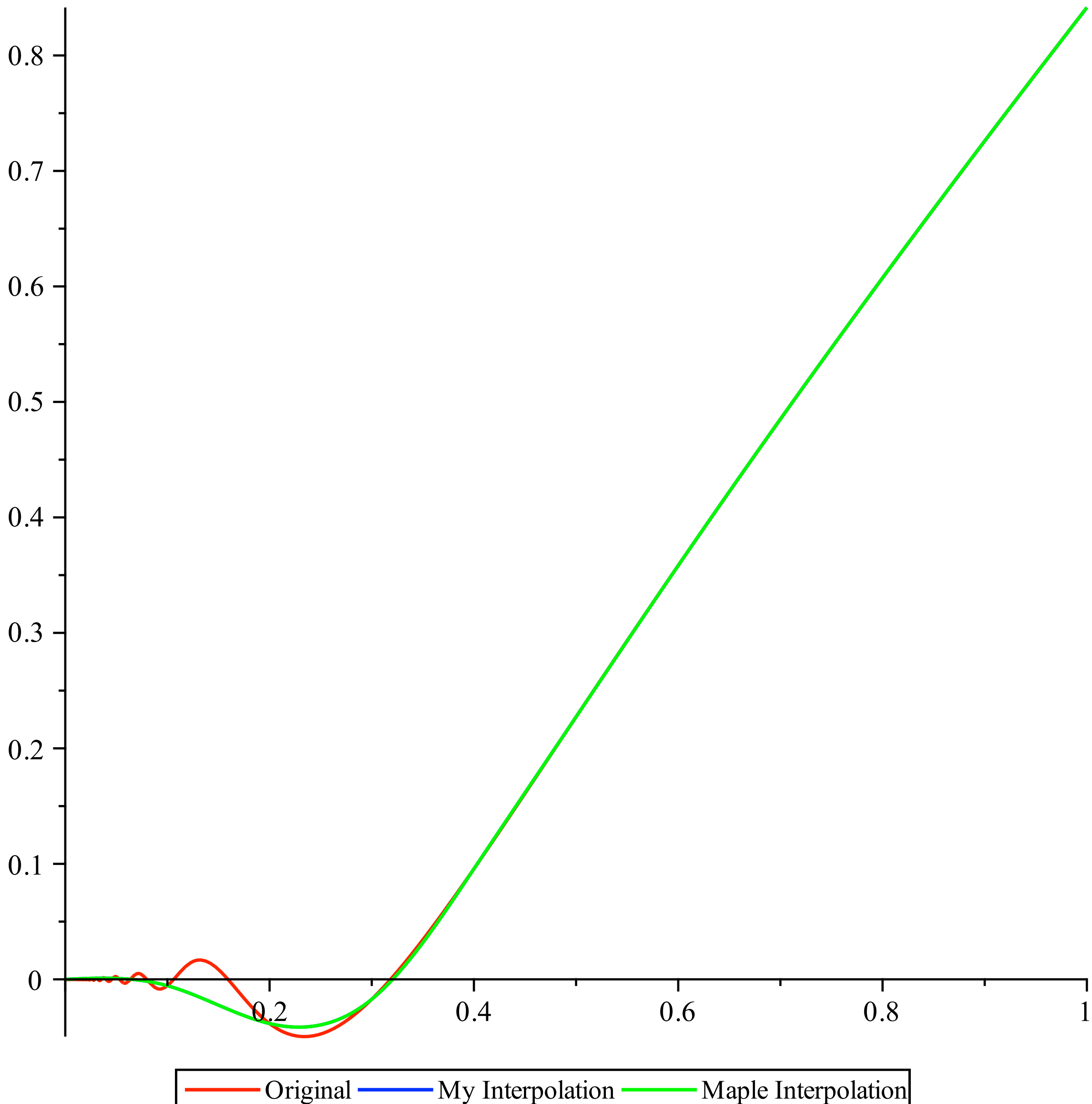
```
print("Cubic splines") ;
plot([f, myCubicSplineInterp, mapleCubicSplineInterp], section, color = [red, blue, green],
```

legend = ["Original", "My Interpolation", "Maple Interpolation"]);
ComputeIterpolationError(*myCubicSplineInterp*, *mapleCubicSplineInterp*, *n*, *section*);

print("B splines") ;
plot([f, myBSplineInterp, mapleBSplineInterp], section, color = [red, blue, green], legend = ["Original", "My Interpolation", "Maple Interpolation"]);

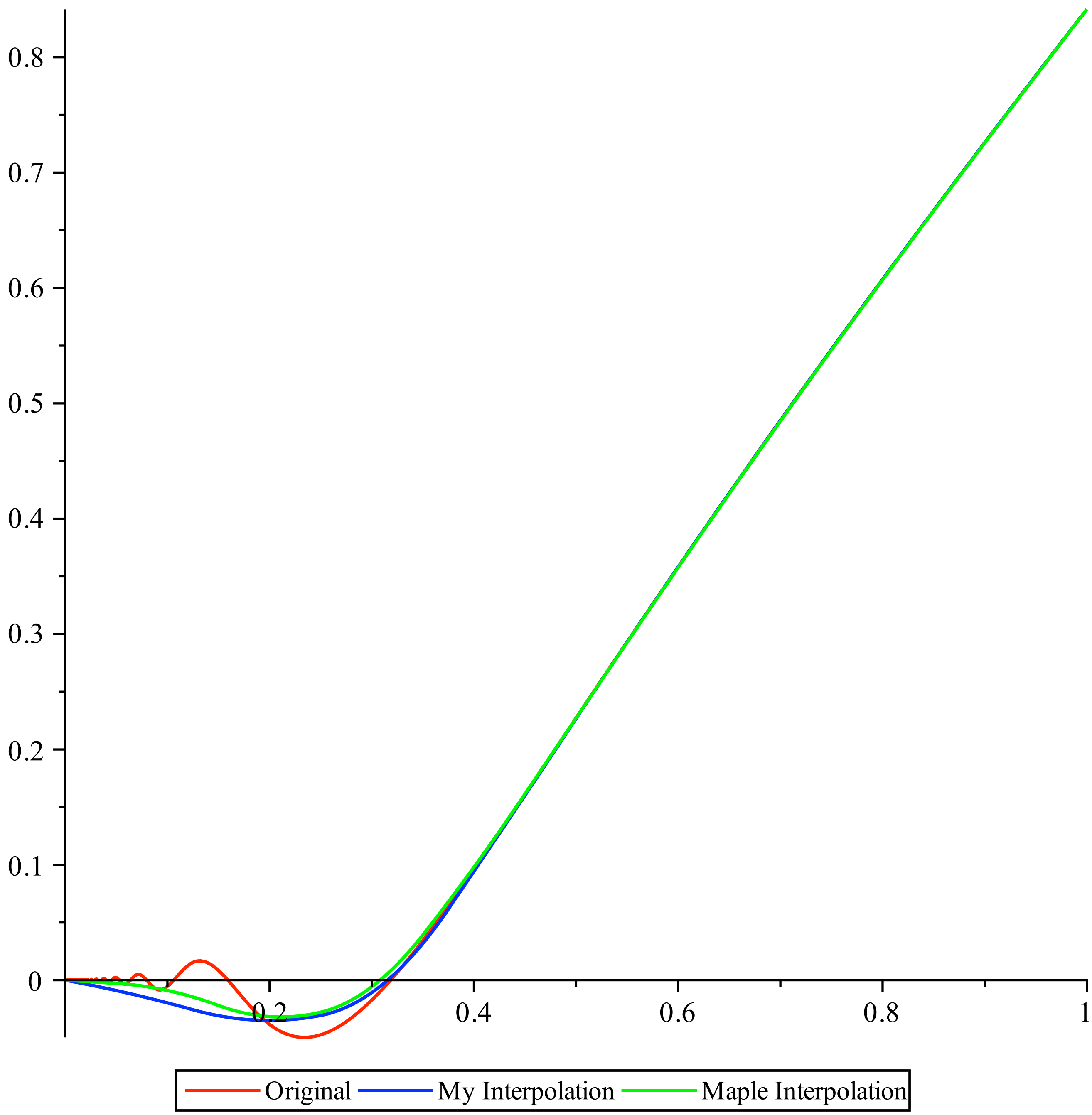
```
Warning, (in mapleCubicSplineInterp) `i` is implicitly declared local
Warning, (in mapleCubicSplineInterp) `i` is implicitly declared local
```

"Cubic splines"



$3.33066907387547 \times 10^{-16}$

"B splines"



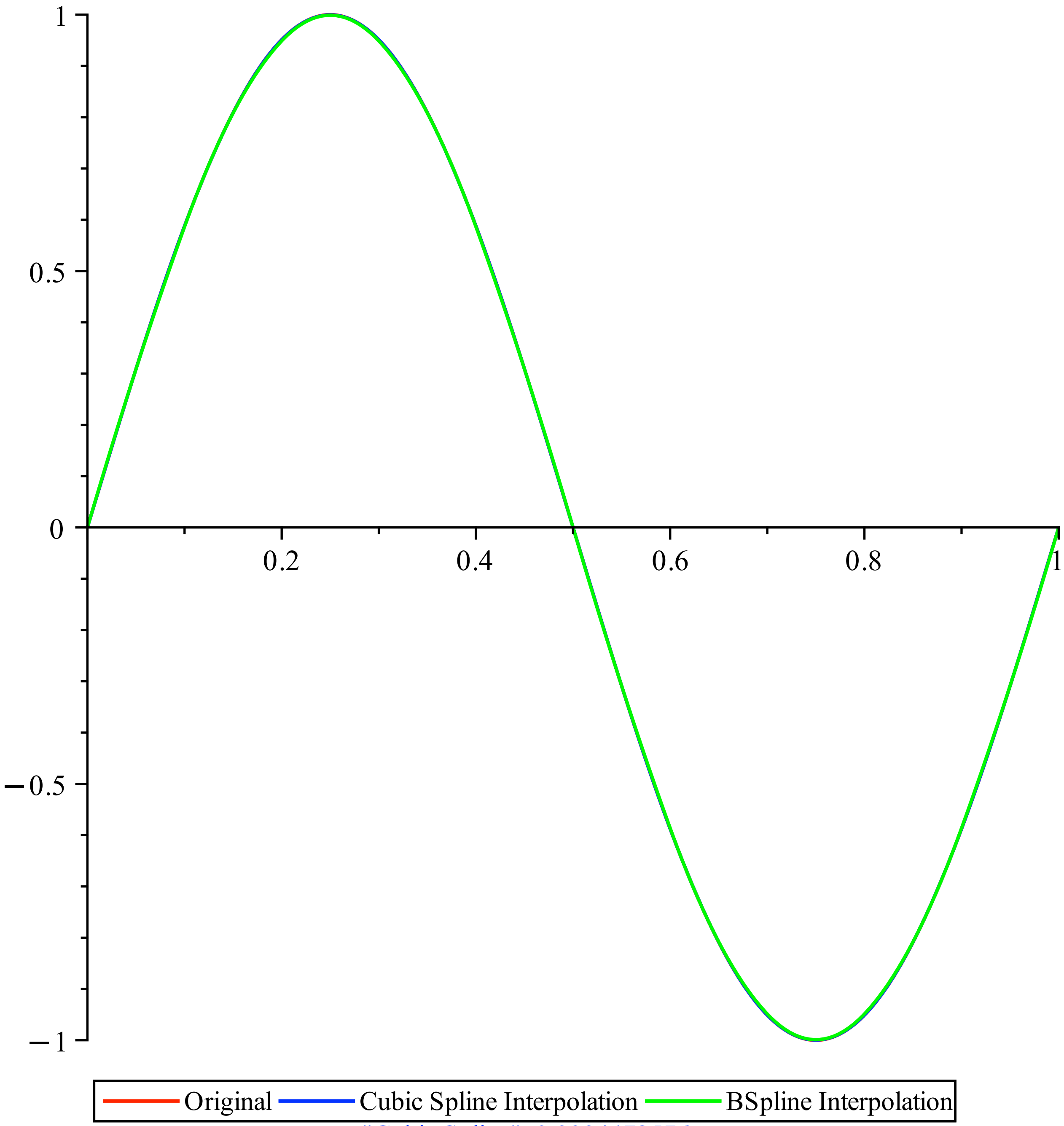
> *# Судя по примеру — сплайны построены корректно. Кубические сплайны практически полностью совпадают с тем, что предоставляет Maple. Построенные B-сплайны отличаются немного от предоставляемых Maple и связано это, скорее всего, с разным выбором коэффициентов k.*

> # *Примеры и эксперименты*

> *# Рассмотрим простую переодическую функцию, которая принадлежит* $C^2[0, 1]$ и *имеет не слишком большой период. Ожидается*, *что как кубические, так* и *квадратичные — сплайны будут хорошо интерполировать эту функцию.*

```
f := x→sin(2 Pi ·x);
myCubicSplineInterpolation := CubicSplineInterpolation(n, section, border1, border2, f) :
myBSplineInterpolation := BSplineInterpolation(n + 2, section, eps, f) :
ShowGraphs(f, myCubicSplineInterpolation, myBSplineInterpolation);
print("Cubic Spline", ComputeIterpolationError(f, myCubicSplineInterpolation, n, section) ) :
print("B-Spline", ComputeIterpolationError(f, myBSplineInterpolation, n, section)) :
```

$$f := x \mapsto \sin(2 \cdot \pi \cdot x)$$



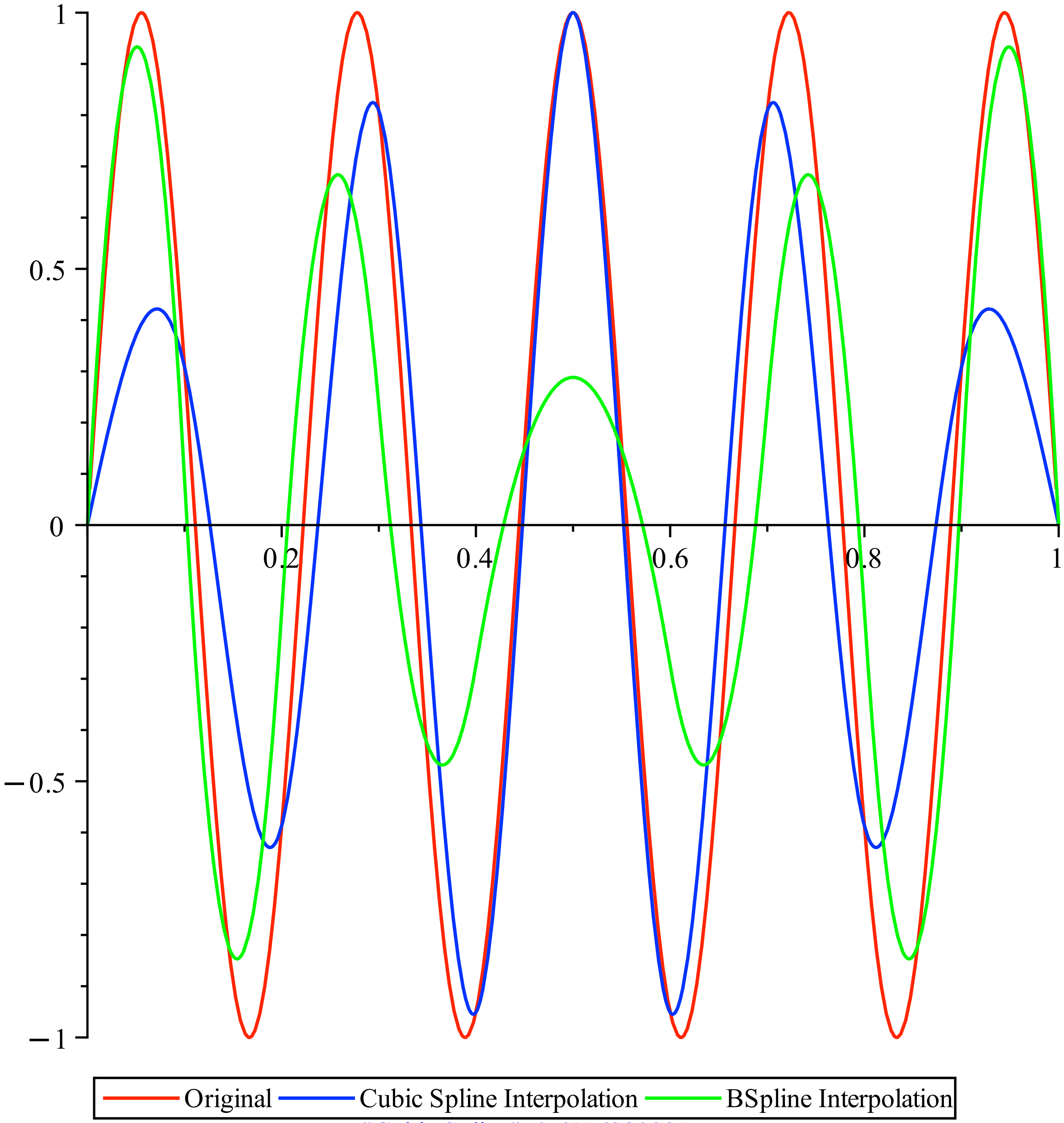
"Cubic Spline", 0.0004472576

"B-Spline", 0.0027470087 **(1)**

> # *Рассмотрим эту же функцию, только значительно увеличим период. Ожидается, что на этот раз, как минимум, кубические сплайны будут плохо интерполировать функцию. Связано это с тем, что сетка слишком крупная и между двумя узлами значения функции слишком сильно несколько раз изменяются. Как можем заметить из графика - квадратичные B-сплайны тоже плохо приблежают "слишком колеблющиеся" функции.*

```
f := x→sin(9 Pi·x);
myCubicSplineInterpolation := CubicSplineInterpolation(n, section, border1, border2, f) :
myBSplineInterpolation := BSplineInterpolation(n + 2, section, eps, f) :
ShowGraphs(f, myCubicSplineInterpolation, myBSplineInterpolation);
print("Cubic Spline", ComputeIterpolationError(f, myCubicSplineInterpolation, n, section) ) :
print("B-Spline", ComputeIterpolationError(f, myBSplineInterpolation, n, section)) :
```

$$f := x \mapsto \sin(9 \cdot \pi \cdot x)$$



"Cubic Spline", 0.6176380995

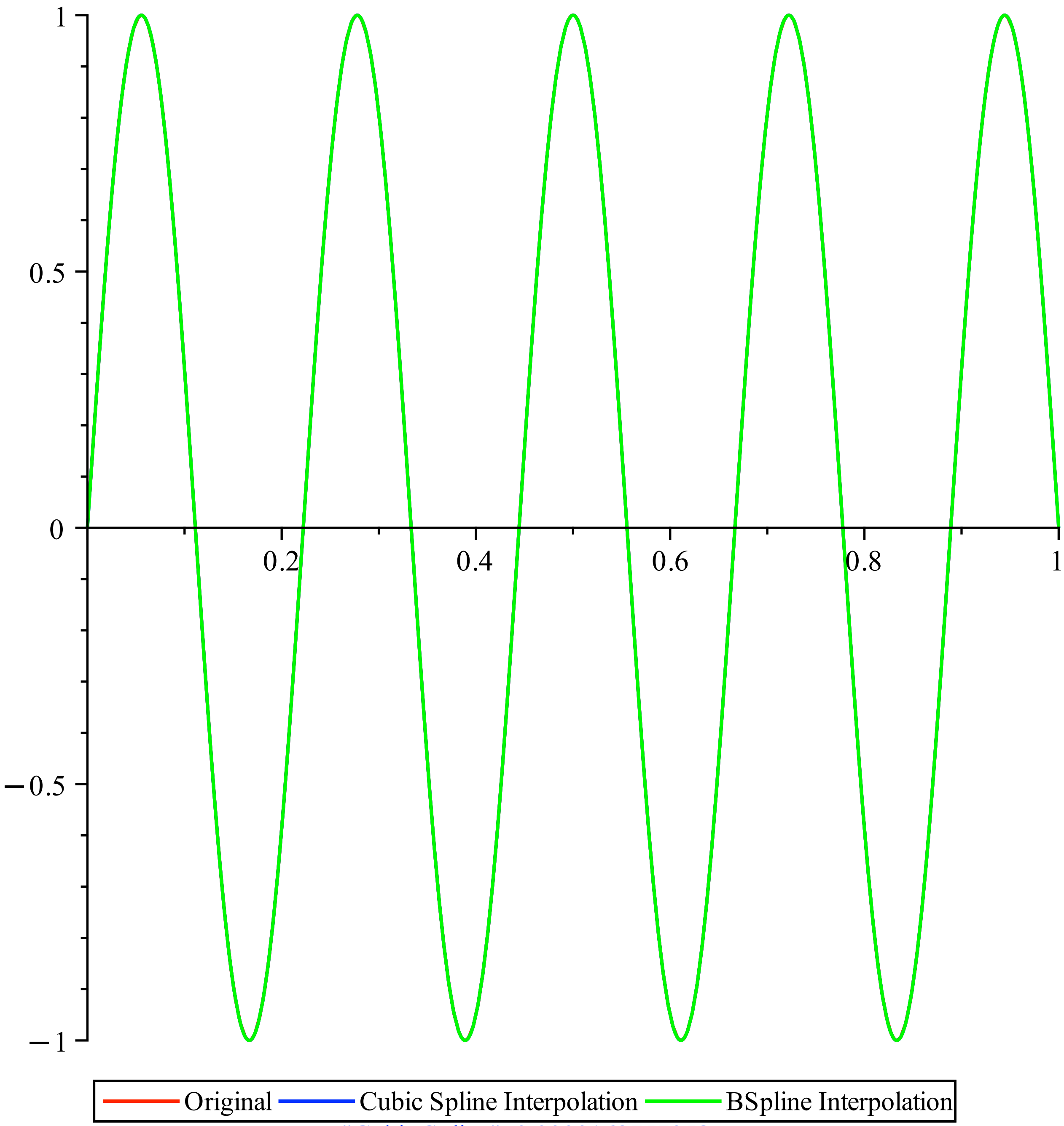
"B-Spline", 0.7116028118 **(2)**

> *#Рассмотрим предыдущий пример и увеличим размер сетки до 100. Ожидается, что теперь кубические сплайны будут интерполировать функцию намного точнее. Из графика видно, что B-сплайны также хорошо справляются с задачей на сетке такого размера.*

```
f := x→sin(9 Pi·x);
myCubicSplineInterpolation := CubicSplineInterpolation(10· n, section, border1, border2, f) :
myBSplineInterpolation := BSplineInterpolation(10·n + 2, section, eps, f) :
ShowGraphs(f, myCubicSplineInterpolation, myBSplineInterpolation);
print("Cubic Spline", ComputeIterpolationError(f, myCubicSplineInterpolation, 10 n,
    section) ) :
```

$print("B\text{-}Spline", ComputeIterpolationError(f, myBSplineInterpolation, 10\, n, section)):$

$$f := x \mapsto \sin(9 \cdot \pi \cdot x)$$



"Cubic Spline", 0.0000169757058

"B-Spline", 0.0001992700 **(3)**

> *#Рассмотрим похожий пример. Ожидания такие же, как и в позапрошлом примере. В местах, где в пределах двух узлов функция изменяется слишком быстро и сильно (в нашем случае это где − то между узлами 0 и 0.1; частично между* 0.1 и 0.2) *кубические сплайны будут плохо интерполировать функцию.* В *местах же, где* в *пределах двух узлов функция изменяется плавно − кубические сплайны будут интерполировать функцию точнее*

. *Как видно по графику — это справедливо* и *для B — сплайнов.*

$$f := x \to \sin\left(\frac{1}{x+\frac{1}{10}}\right);$$

$myCubicSplineInterpolation := CubicSplineInterpolation(n, section, border1, border2, f)$:

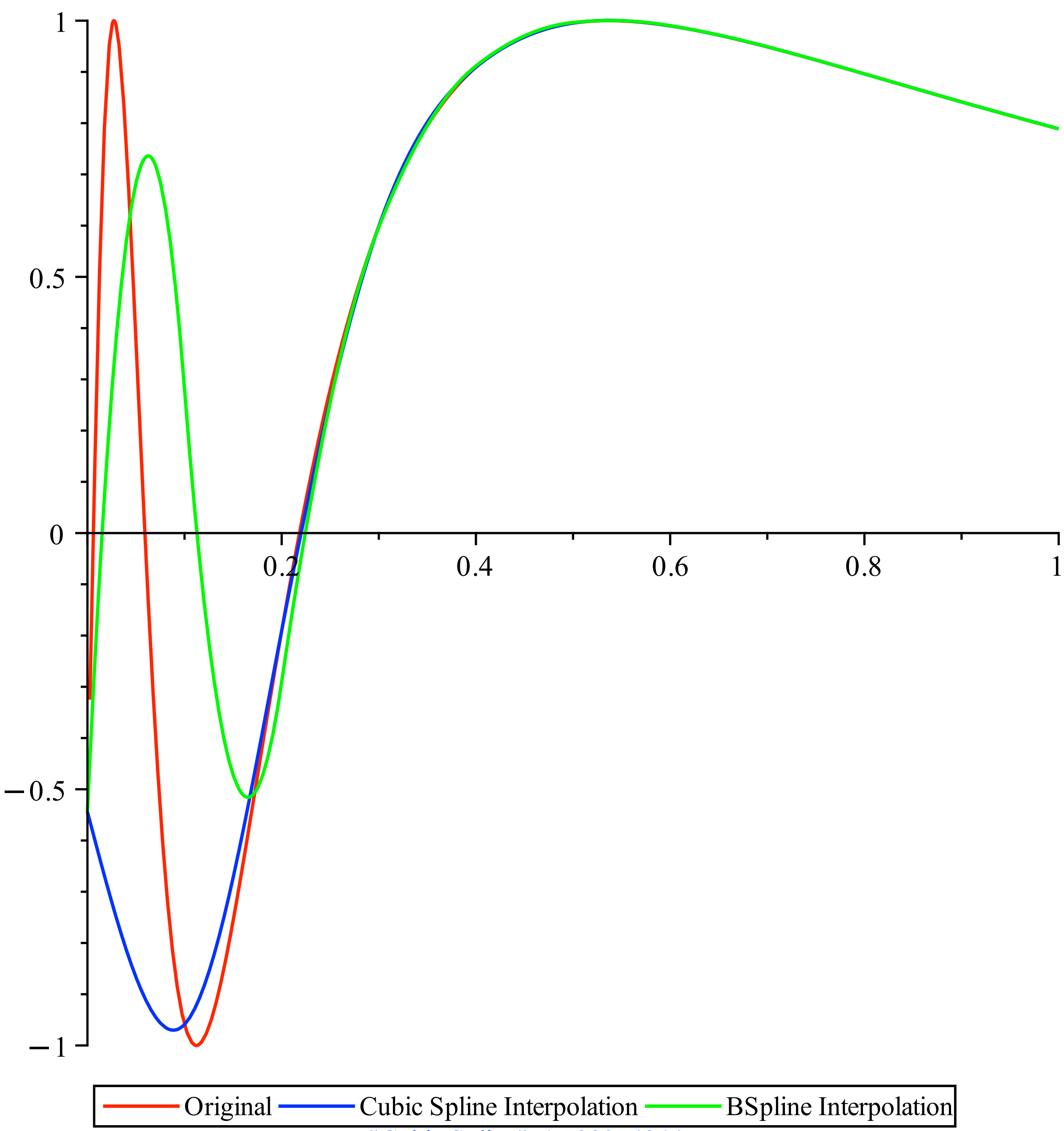
$myBSplineInterpolation := BSplineInterpolation(\mathrm{n}+2, \mathrm{section}, \mathrm{eps}, f)$:

$ShowGraphs(f, myCubicSplineInterpolation, myBSplineInterpolation)$;

$print($"Cubic Spline", $ComputeIterpolationError(f, myCubicSplineInterpolation, n, section))$:

$print($"B-Spline", $ComputeIterpolationError(f, myBSplineInterpolation, n, section))$:

$$f := x \mapsto \sin\left(\frac{1}{x+\frac{1}{10}}\right)$$



"Cubic Spline", 1.739374844

"B-Spline", 1.343812779 **(4)**

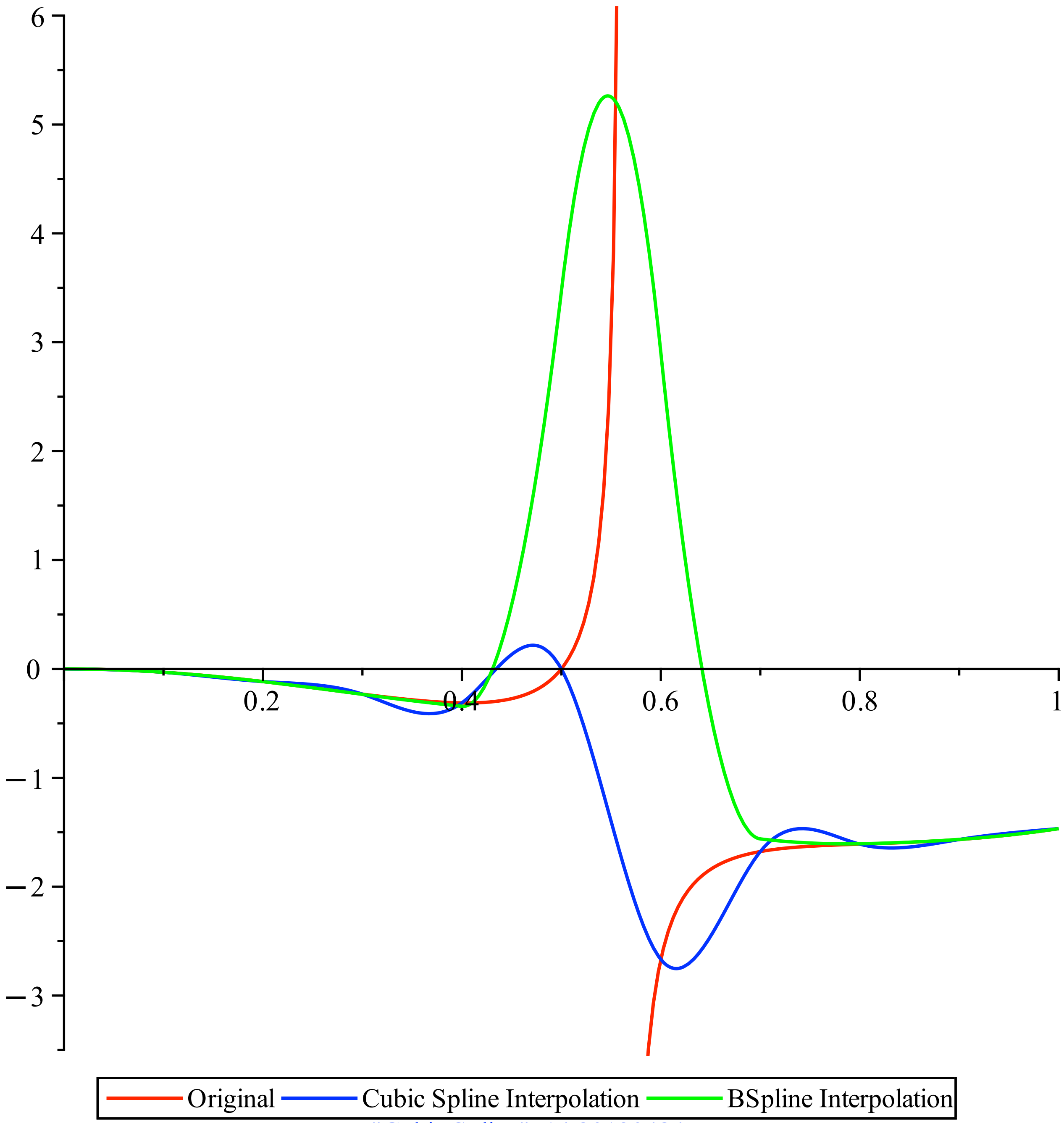
>

> *#Использование интерполяции сплайнами адекватно только тогда, когда* $f \in C^2[a, b]$ *. Рассмотрим функцию, которая разрывна на отрезке* $[0, 1]$ в *точке* $\sqrt{\frac{1}{\pi}}$ *. Ождиается, что как кубические сплайны, так* и *B* − *сплайны будут очень плохо её интепролировать.*

$$f := x \rightarrow \frac{x^2 \cos(\mathrm{Pi} \cdot x)}{x^2 - \frac{1}{\mathrm{Pi}}};$$

$myCubicSplineInterpolation := CubicSplineInterpolation(n, section, border1, border2, f)$:
$myBSplineInterpolation := BSplineInterpolation(\mathrm{n} + 2, \mathrm{section}, \mathrm{eps},\ f)$:
$ShowGraphs(f, myCubicSplineInterpolation, myBSplineInterpolation)$;
$print(\text{"Cubic Spline"}, ComputeIterpolationError(f, myCubicSplineInterpolation, n, section))$:
$print(\text{"B-Spline"}, ComputeIterpolationError(f, myBSplineInterpolation, n, section))$:

$$f := x \mapsto \frac{x^2 \cdot \cos(\pi \cdot x)}{x^2 - \frac{1}{\pi}}$$



"Cubic Spline", 14.20130484

"B-Spline", 15.56497883 **(5)**

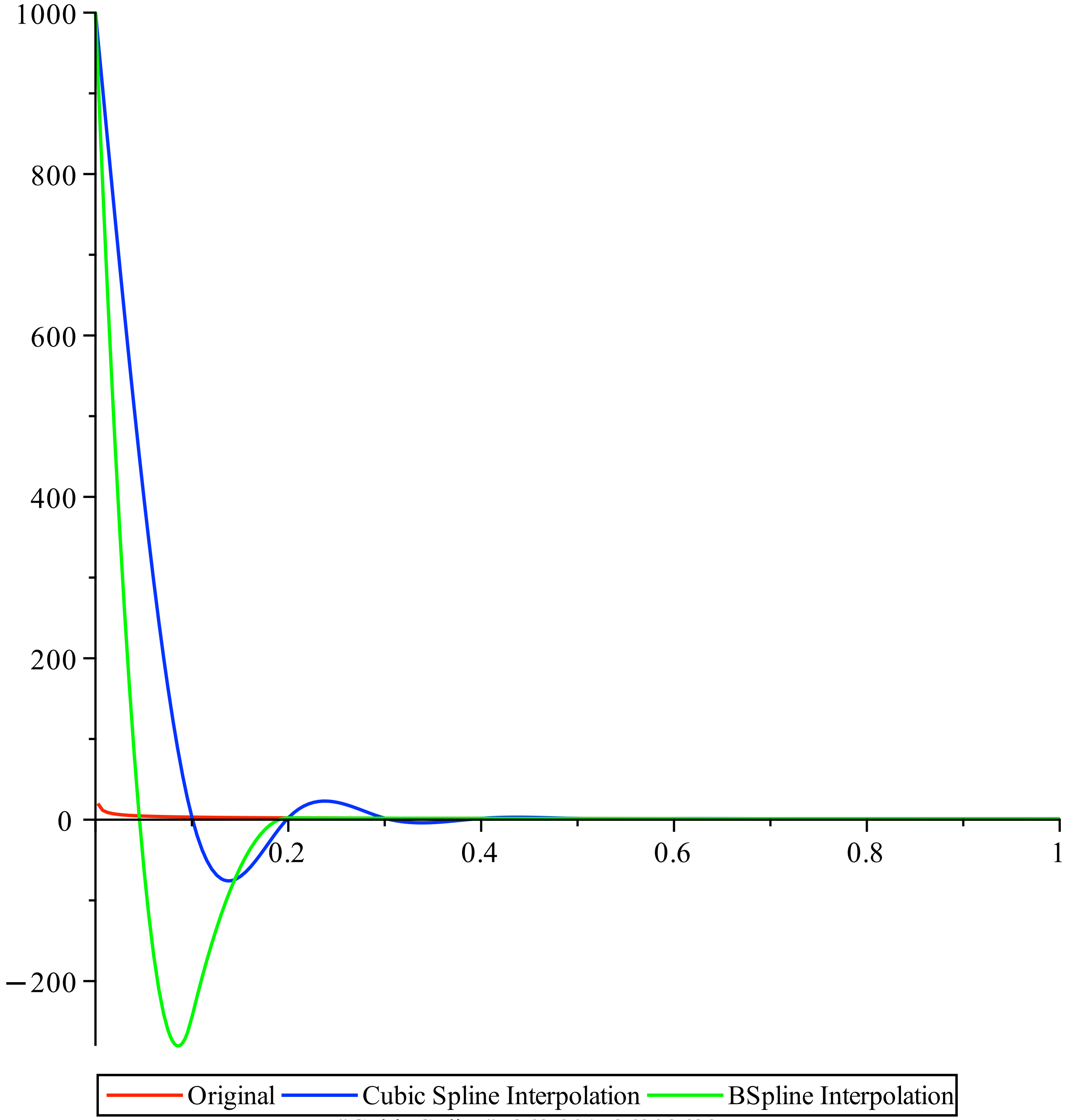
> #*Рассмотрим функцию, у которой разрыв происходит не на отрезке [0,1].*

$$f := x \rightarrow \frac{1}{\sqrt{x + 0.000001}};$$

myCubicSplineInterpolation := *CubicSplineInterpolation*(*n*, *section*, *border1*, *border2*, *f*) :
myBSplineInterpolation := *BSplineInterpolation*(n + 2, section, eps, *f*) :
ShowGraphs(*f*, *myCubicSplineInterpolation*, *myBSplineInterpolation*);
print("Cubic Spline", *ComputeIterpolationError*(*f*, *myCubicSplineInterpolation*, *n*, *section*)) :
print("B-Spline", *ComputeIterpolationError*(*f*, *myBSplineInterpolation*, *n*, *section*)) :

 # Разрыв происходит очень близко к нулю, поэтому в его окрестности как кубические сплайны, так и квадратичные B−сплайны плохо интерполируют функцию.

$$f := x \mapsto \frac{1}{\sqrt{x + 1. \times 10^{-6}}}$$



"Cubic Spline", 863.901596309623

"B-Spline", 709.0774755 **(6)**

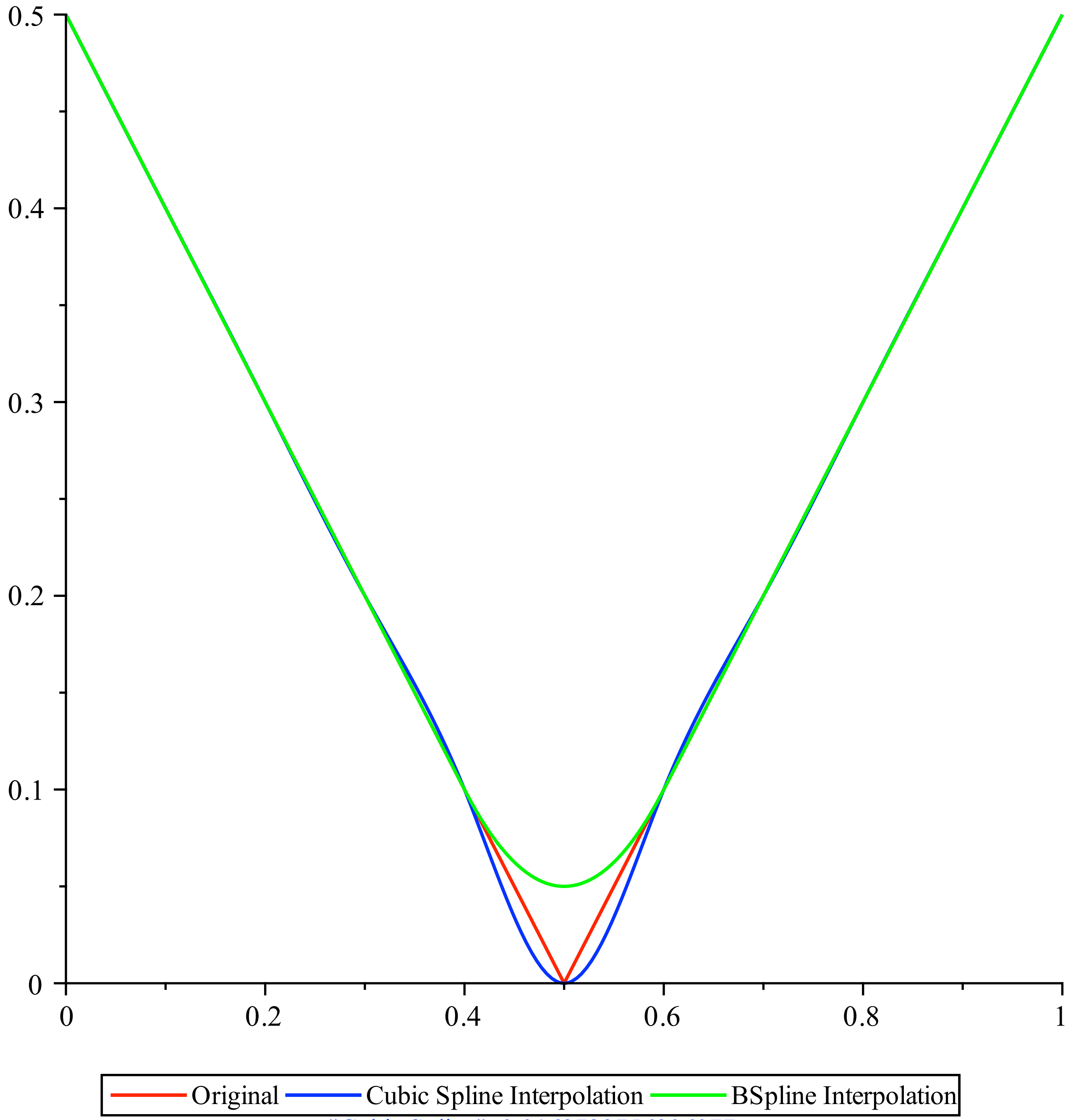
>

> *#Рассмотрим функцию, у которой в точке 0.5 первая производная <u>не будет существовать</u>. Ождитается, что в таком случае, в окрестности точки 0.5 сплайны (как минимум кубические) будут плохо приблежать исходную функцию (по тем же соображениями, что и в прошлом примере).*

```
f := x → abs(x − 0.5);
myCubicSplineInterpolation := CubicSplineInterpolation(n, section, border1, border2, f) :
myBSplineInterpolation := BSplineInterpolation(n + 2, section, eps, f) :
ShowGraphs(f, myCubicSplineInterpolation, myBSplineInterpolation);
```

```
print("Cubic Spline", ComputeIterpolationError(f, myCubicSplineInterpolation, n, section)) :
print("B-Spline", ComputeIterpolationError(f, myBSplineInterpolation, n, section)) :
```

$$f := x \mapsto |x - 0.5|$$



"Cubic Spline", 0.0169723756906077

"B-Spline", 0.05000000000 **(7)**

> *# Рассмотрим непрерывную функцию, у которой первая производная имеет разрыв в точке 0.5. Ожидается, что в окрестности этой точки кубические сплайны будут плохо интерполировать функцию. Из графика видно, что это справедливо и для квадратичных B-сплайнов.*

$f := x \rightarrow piecewise\left(0 \leq x < \frac{1}{2}, 2 \cdot x^2, \frac{1}{2} \leq x \leq 1, 1 - x\right);$

$myCubicSplineInterpolation := CubicSplineInterpolation(n, section, border1, border2, f) :$

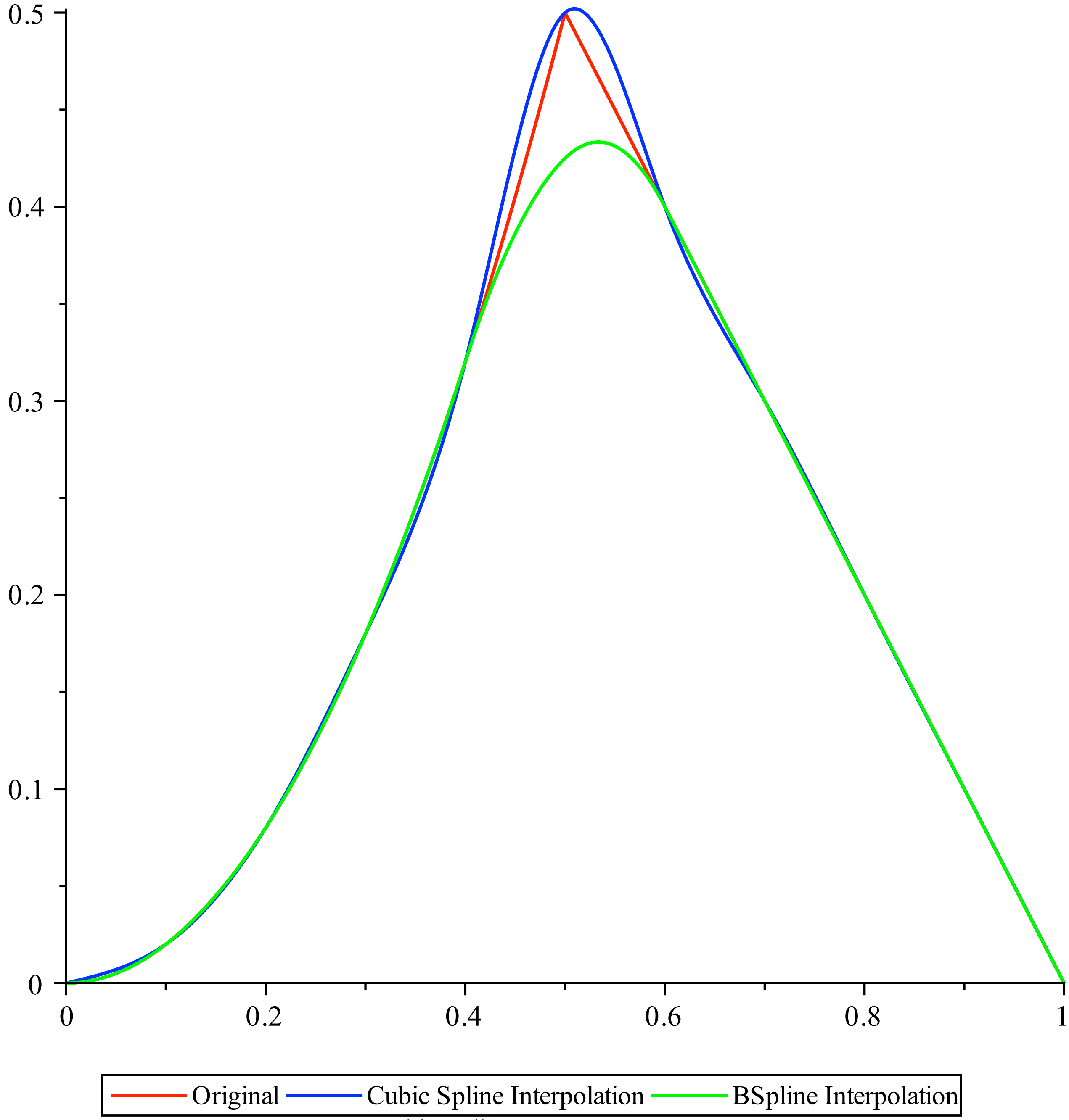
$myBSplineInterpolation := BSplineInterpolation(\mathrm{n} + 2, \mathrm{section}, \mathrm{eps}, f) :$

$ShowGraphs(f, myCubicSplineInterpolation, myBSplineInterpolation);$

$print($"Cubic Spline"$, ComputeIterpolationError(f, myCubicSplineInterpolation, n, section)) :$

$print($"B-Spline"$, ComputeIterpolationError(f, myBSplineInterpolation, n, section)) :$

$$f := x \mapsto \begin{cases} 2 \cdot x^2 & 0 \leq x < \frac{1}{2} \\ 1 - x & \frac{1}{2} \leq x \leq 1 \end{cases}$$



"Cubic Spline", 0.02644647863

"B-Spline", 0.07500000000 **(8)**

> *#Зададим кусочно-непрерывную, монотонную, всюду неотрицательную на отрезке* [0, 1] *функцию. Построим интерполяцию кубическими сплайнами.*

$f := x \rightarrow piecewise(x < 0.4, 0, x - 0.4);$

$myCubicSplineInterpolation := CubicSplineInterpolation(n, section, border1, border2, f):$

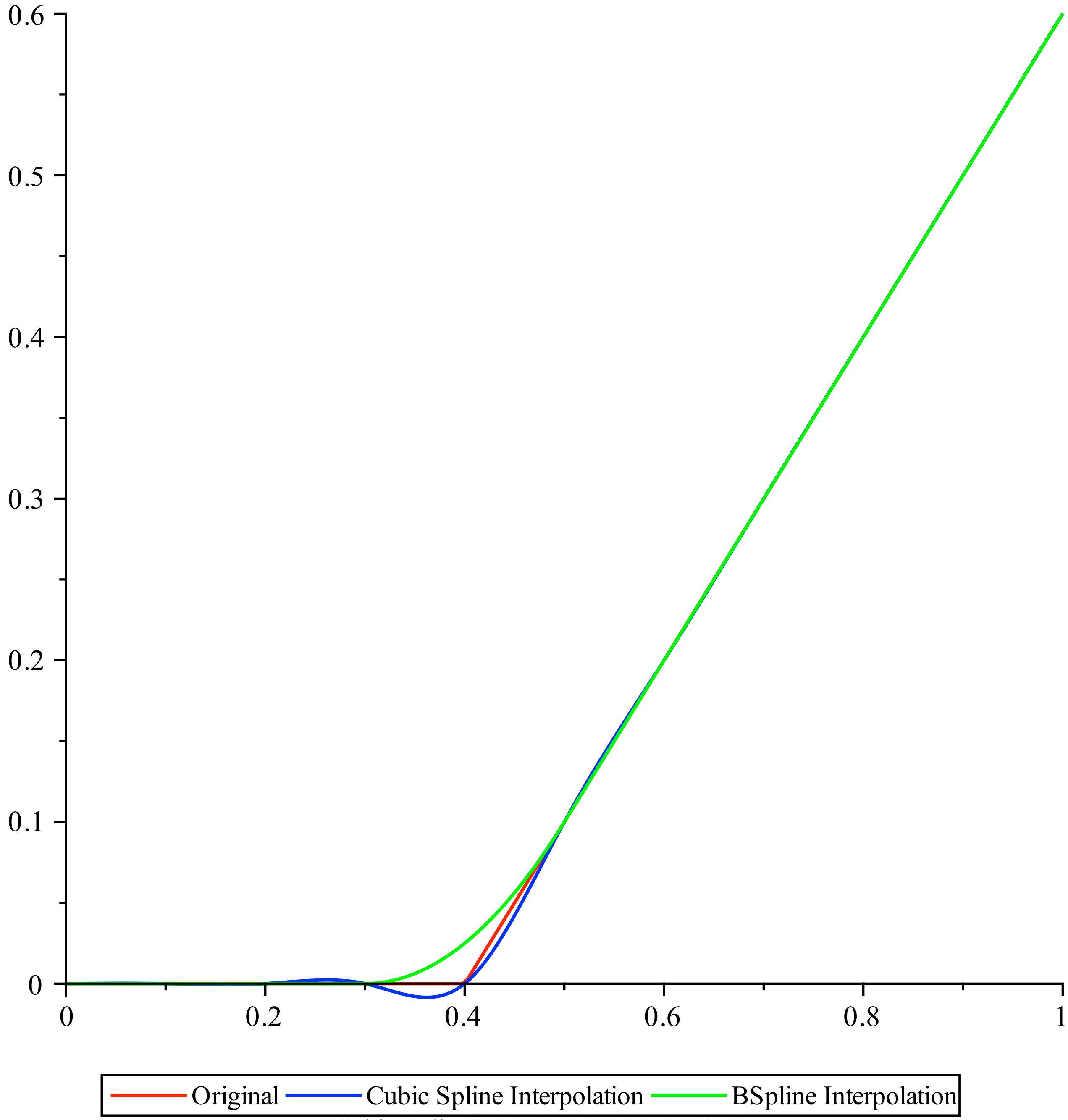
$myBSplineInterpolation := BSplineInterpolation(\mathrm{n} + 2, \mathrm{section}, \mathrm{eps}, f):$

$ShowGraphs(f, myCubicSplineInterpolation, myBSplineInterpolation);$

$print(\text{"Cubic Spline"}, ComputeIterpolationError(f, myCubicSplineInterpolation, n, section)):$

$print(\text{"B-Spline"}, ComputeIterpolationError(f, myBSplineInterpolation, n, section)):$

$$f := x \mapsto \begin{cases} 0 & x < 0.4 \\ x - 0.4 & \mathit{otherwise} \end{cases}$$



"Cubic Spline", 0.00848682227920378

"B-Spline", 0.02500000000 **(9)**

> *# Как можно заметить, у полученной интерполяции нарушается монотонность и неотрицательность. Оно и ожидаемо − как можно заметить по функции, в узле 0.4 нарушается равенство первых производных. B−сплайны в данном примере не нарушают свойство неотрицательности, но также плохо интерполируют функцию в окрестности узла 0.4.*

> *# Мини-результат: во всех примерах ожидания оправдались.*