SPLINE INTERPOLATION

Q 1 : CONVERTING THE FUNCTION TO A TABLE

**from** scipy **import** **\***

**from** matplotlib**.**pyplot **import** **\***

**import** os

**import** re

**import** scipy**.**integrate **as** intg

**import** scipy**.**special **as** sp

**def** f**(**x**):**

num **=** **pow(**x**,**1**+**sp**.**jn**(**0**,**x**))**

densqr**=** **(**1**+**100**\***x**\*\***2**)\*(**1**-**x**)**

den**=**sqrt**(**densqr**)**

**return** num**/**den

x**=**arange**(**0.1**,**0.95**,**0.05**)**

y**=**f**(**x**)**

**for** i **in** **range(**0**,**17**,**1**):**

**print(**x**[**i**],**end **=**" "**)**

**print(**y**[**i**])**

figure**(**1**)**

plot**(**x**,**y**)**

xlabel**(**"Location(x)"**)**

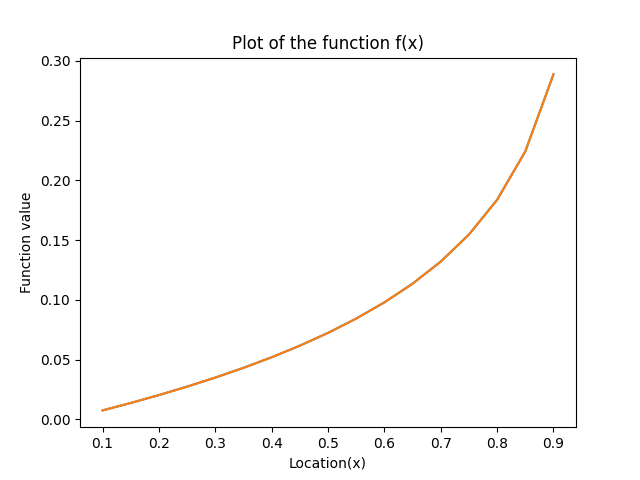
ylabel**(**"Function value"**)**

title**(**"Plot of the function f(x)"**)**

savefig**(**"SplineQ1.png"**)**

|  |  |
| --- | --- |
| 0.1 | 0.007496562832422993 |
| 0.15000000000000002 | 0.013682307550529474 |
| 0.20000000000000004 | 0.020323674974640826 |
| 0.25000000000000006 | 0.027387405159075735 |
| 0.30000000000000004 | 0.03494557972740577 |
| 0.3500000000000001 | 0.04309509544278532 |
| 0.40000000000000013 | 0.05194921947898973 |
| 0.45000000000000007 | 0.06164521667354194 |
| 0.5000000000000001 | 0.0723587017455112 |
| 0.5500000000000002 | 0.0843259149245415 |
| 0.6000000000000002 | 0.0978798407477116 |
| 0.6500000000000001 | 0.1135128247384625 |
| 0.7000000000000002 | 0.13199353669417332 |
| 0.7500000000000002 | 0.15460593998515973 |
| 0.8000000000000002 | 0.18369806957867635 |
| 0.8500000000000002 | 0.22416760626939342 |
| 0.9000000000000002 | 0.28865914483840577 |

Q 2 : PLOTTING THE FUNCTION



The function is analytic in the region [0.1,0.9] since its derivative exists and is continuous on this interval. The function has one singularity when defined on the real numbers ℝ: at x=1 and two additional singularities when defined on the complex numbers ℂ: at x=±0.1i. Thus, the radius of convergence of the function is 0.1 at both x=0.1 and x=0.9 .

Q 3: Vary the number of uniformly spaced spline points with boundary condition of y’’=0

**from** scipy **import** **\***

**from** scipy **import** special

**from** matplotlib**.**pyplot **import** **\***

**import** spline

**def** func**(**xa**):**

num**=pow(**xa**,**1**+**special**.**jn**(**0**,**xa**))**

densqr**=(**1**+**100**\***xa**\***xa**)\*(**1**-**xa**)**

den**=**sqrt**(**densqr**)**

**return** num**/**den

h**=**logspace**(-**4**,-**2**,**20**)**

N**=(**0.8**)/**h

err**=**zeros**(**h**.**shape**)**

figure**(**0**)**

**for** i **in** **range(len(**h**)):**

n**=int(**N**[**i**])**

xa**=**linspace**(**0.1**,**0.90**,** **int(**N**[**i**]))**

ya**=**func**(**xa**)**

y2a**=**zeros**(**ya**.**shape**)**

xx**=**linspace**(**0.1**,**0.9**,**10**\***n**+**1**)**

yt**=**func**(**xx**)**

yy**=**zeros**(**xx**.**shape**)**

u**=**zeros**(**ya**.**shape**)**

y2a**=**spline**.**spline**(**xa**,**ya**,**1e40**,**1e40**)**

yyb**=**spline**.**splintn**(**xa**,**ya**,**y2a**,**xx**)**

**if** i**==**0 **:**

figure**(**1**)**

plot**(**xa**,**ya**,**'ro'**,**xx**,**yt**,**'b+'**,**xx**,**yyb**,**'k'**)**

#ylim(0,0.3)

title**(**"Interpolated values and data points for n=%d" **%** N**[**i**])**

legend**([**"ya"**,**"yt"**,**"yyb"**])**

grid**(True)**

savefig**(**"Figure4.png"**)**

figure**(**0**)**

z**=abs(**yyb**-**func**(**xx**))**

plot**(**xx**,**z**,** linewidth**=**0.5**,** linestyle**=**"-"**,**label**=**"N=%d"**%**N**[**i**])**

err**[**i**]=**z**.max()**

xlim**(**0.7**,**0.9**)**

xlabel**(**"Location (x)"**)**

ylabel**(**"Error profile"**)**

rc**(**'legend'**,** fontsize**=**8**)**

grid**(True)**

savefig**(**"Figure5.png"**)**

**print(**err**)**

legend**(**loc**=**"upper left"**)**

figure**(**1**)**

loglog**(**h**,**err**)**

xlim**(**0.0001**,**0.01**)**

ylim**(**0**,**0.001**)**

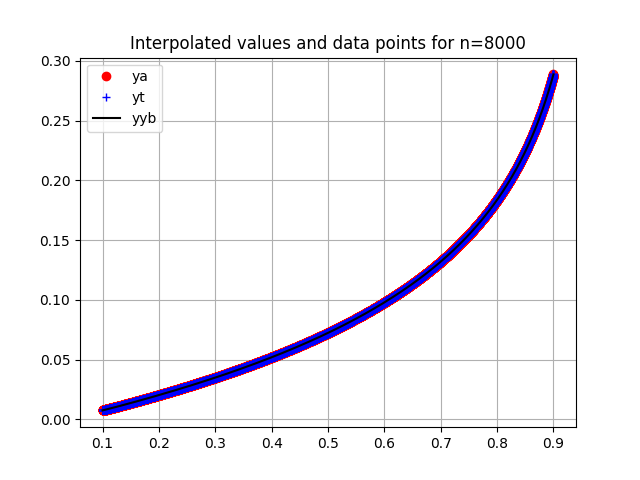
xlabel**(**"Spacing"**)**

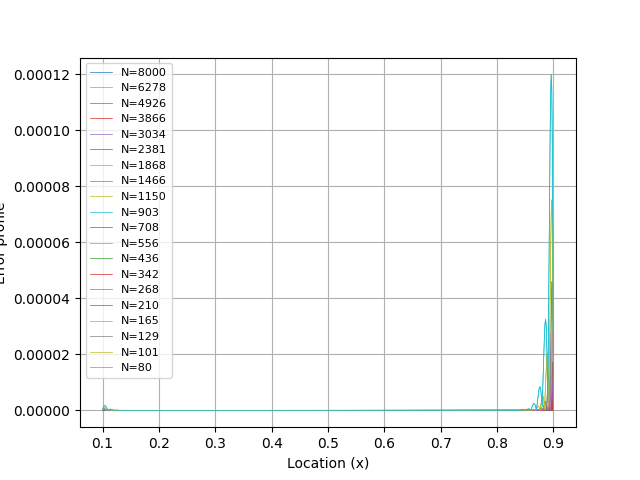
ylabel**(**"Error"**)**

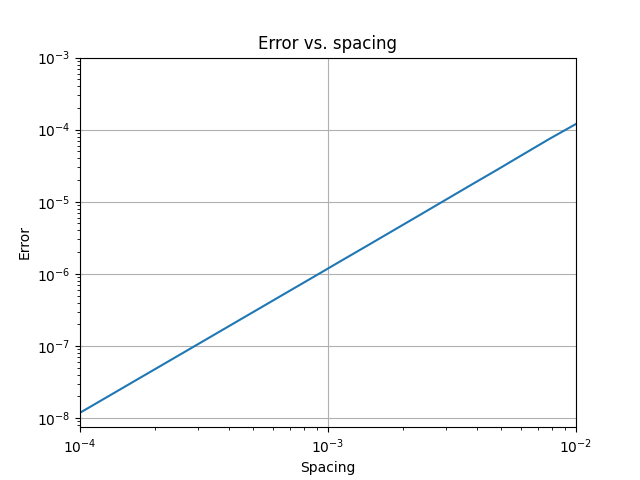
title**(**"Error vs. spacing"**)**

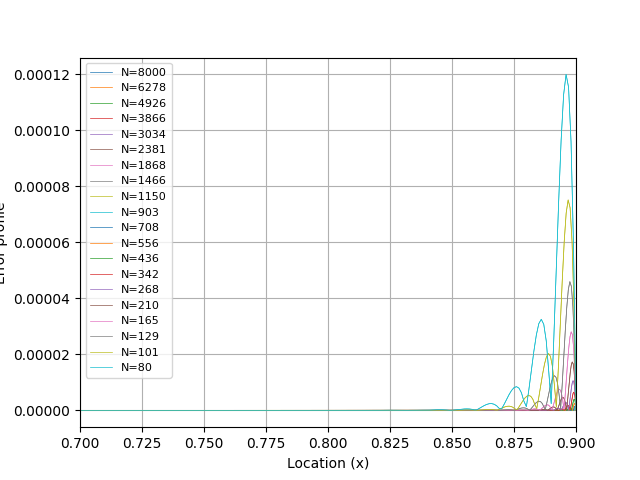
grid**(True)**

savefig**(**"Figure6.png"**)**



****

****

****

From the above results we can conclude that the larger the number of points (and hence smaller the spacing between them) being evaluated, the smaller the error becomes. For obtaining an accuracy to the sixth decimal place, the error should be below 5x10 − 7 and from the graph above, we can see that we need a spacing of nearly 10-4 , or N=8000.