درونیابی اسپلاین مکعبی

و مقایسه ای با دیگر روش ها

زمستان ۱۴۰۲

دانشگاه صنعتی شریف

محمد حسن شیری

دانشکده فیزیک

مقدمه:

در علوم ریاضی و مهندسی، اغلب ما با داده‌های ناقص یا محدود روبرو هستیم. توابعی که اطلاعات ما را توصیف می‌کنند، نمونه‌های معین و محدودی را در اختیار دارند ولی برخی اوقات ما به دنبال یک نمایش یا تقریب یکنواخت این اطلاعات در کل بازه‌ها هستیم. در اینجا به وسیله روش‌های تقریب و انتقال، ما سعی می‌کنیم این وظیفه را انجام دهیم.

یکی از مهمترین اهداف این کار، افزایش دقت در پیش‌بینی و تخمین توابع است. زمانی که ما با توابع پیچیده یا داده‌های نویزی سروکار داریم، معمولاً دشوار است تا توابع را به دقت پیش‌بینی کنیم. به کمک روش‌های تقریب، ما می‌توانیم توابع پیچیده را به صورت تقریبی با توجه به اطلاعات موجود در داده‌های محدود تخمین بزنیم.

مفهوم تکنیک‌های تقریب و انتقال در مطالعه علوم مختلف بسیار مهم است. متدهای انتقال و تقریب به ما کمک می‌کنند تا اطلاعات بین نقاط داده‌های شناخته‌شده را با دقت بیشتری پیش‌بینی کرده و اطلاعات جدید را از طریق این تقریب‌ها استخراج کنیم. در این مسیر، ما از مدل‌ها و الگوریتم‌های مختلفی برای تقریب توابع یا داده‌های غیرمنظم استفاده می‌کنیم.

در مثالهایی که ارائه داده‌ایم، ما از روش‌های مختلفی مانند انتقال مکعبی (Cubic Spline)، انتقال درجه دوم (Quadratic Spline)، و انتقال نیوتنی (Newtonian Interpolation) برای تقریب توابع متعالی معروف مانند sin(x)، e^x و cosh(x) استفاده کردیم. این روش‌ها به ما امکان می‌دهند تا از اطلاعات موجود در نقاط داده‌ها به صورت پیوسته به دیگر نقاط مستقل از آنها تعمیم دهیم و توابع را در بازه‌های بین داده‌های ما تخمین بزنیم.

با استفاده از این تقریب‌ها، می‌توانیم به سادگی مواردی مانند افزایش دقت در پیش‌بینی یا تخمین توابع در نقاطی که داده‌ها موجود نیستند، را دنبال کنیم. این روش‌ها در انواع حوزه‌ها از علوم ریاضی گرفته تا مهندسی و علوم داده به کار می‌روند و برای حل مسائل مختلفی از تحقیقات تا کاربردهای عملی بسیار مفیدند.

تئوری و منطق پیاده سازی:

در روش‌های تقریب، پیاده‌سازی به وسیله زبان‌های برنامه‌نویسی و کتابخانه‌های مربوط به آنها انجام می‌شود. در اینجا، از زبان برنامه‌نویسی Python و کتابخانه‌های محاسبات علمی معروف مانند NumPy و SciPy برای پیاده‌سازی و تست روش‌های تقریب استفاده شده است.

در ابتدا، داده‌های نمونه برای توابع مورد نظر تولید شده‌اند. برای مثال، برای تابع sin(x) از توابع NumPy برای تولید نقاط از بازه مورد نظر استفاده شده است. سپس با استفاده از روش‌های تقریبی مانند روش مکعبی، روش درجه دوم، و تقریب نیوتنی، تخمین توابع بر اساس داده‌های نمونه انجام شده است.

پس از پیاده‌سازی روش‌های تقریب، یک تابع جداگانه برای مقایسه عملکرد این روش‌ها نوشته شده است. این تابع با گرفتن داده‌های نمونه و انجام تقریب با هر سه روش، زمان اجرای هر روش را اندازه‌گیری و با هم مقایسه می‌کند.در نهایت، با استفاده از توابع نمایش گرافیکی مانند `matplotlib`، نتایج تقریب رسم شده‌اند. این گرافیک‌ها اطلاعاتی را از داده‌های نمونه، توابع اصلی، و تخمین‌های به دست آمده نمایش می‌دهند.

برای تست‌ها، توابع نمونه متفاوتی مورد استفاده قرار گرفته‌اند از جمله sin(x)، e^x، cosh(x). این انتخاب‌ها به ما امکان تست کارایی روش‌های تقریب در شرایط مختلف را می‌دهد و به خوبی نشان می‌دهد که هر روش در چه مواردی موثرتر است.این انتخاب با توجه به نداشتن دیتاست معنی دار و مشخصی انجام شده به صورتی که صرفا توابعی

که رفتاری ناشناخته و یا نامتعارف دارند بررسی شده تا درصورت استفاده بر روی دیتاستی مشخص در صورت نیاز به ارائه مدل درونیابی دقیقی بتوان انتخابی بهینه تر داشت.

در مرحله نهایی برای مقایسه کارایی روش‌های تقریب، از زمان‌سنج `timeit` در Python استفاده شده است. این ابزار به ما این امکان را می‌دهد که زمان اجرای یک قطعه کد را اندازه‌گیری کرده و از آن برای مقایسه عملکرد روش‌های مختلف استفاده کنیم.تابع `timeit` از کتابخانه `timeit` در Python برای اندازه‌گیری زمان اجرای یک دسته از دستورات استفاده می‌شود. در مثال مربوط به مقایسه عملکرد روش‌های تقریب، تابع `compare_performance` برای اندازه‌گیری زمان اجرای سه روش نوشته شده است.

تابع `compare_performance` به تابع‌های تقریب مورد نظر داده‌ها را می‌دهد و زمان اجرای هرکدام را اندازه‌گیری می‌کند. سپس این زمان‌ها را با یکدیگر مقایسه کرده و نتایج را برمی‌گرداند. در اینجا ما از سه روش درونیابی نیوتونی،اسپلاین مکعبی و اسپلاین مربعی دیگری (به عنوان مثال یک روش تقریبی دیگر) استفاده شده است. این تابع نتایج را به عنوان یک tuple بازمی‌گرداند که در آن زمان اجرای هر روش به ترتیب آورده شده است.

بررسی روش ها از نظر تئوری:

تعاریف و مقایسه بین سه روش متداول تقریب توسط اسپلاین (اسپلاین مکعبی، اسپلاین درجه دوم، و اسپلاین نیوتنی) می‌تواند براساس مزایا و معایب هرکدام انجام شود.

- **اسپلاین مکعبی:**
  - *مزایا:*
    - دقت بالا: این روش با استفاده از پاره‌توابع مکعبی برای هر بازه، توانایی ارائه تقریبی دقیق از داده‌ها را دارد.
    - پیوستگی: اسپلاین مکعبی در نقاط انتهایی هر بازه پیوسته است.
    - قابلیت استفاده در انواع توابع: برای تقریب توابع پیوسته و صعودی، این روش عملکرد بسیار خوبی دارد.
  - *معایب:*
    - پیچیدگی: محاسبه و نگهداری ضرایب برای هر بازه می‌تواند محاسباتی پیچیده به همراه داشته باشد.
    - اضافه‌وزن در نقاط انتهایی: در توابع پیوسته، این روش ممکن است به دلیل اضافه‌وزن در نقاط انتهایی بازه‌ها، به نتایجی مناسب نرسد.

- **اسپلاین درجه دوم (کوادراتیک):**
  - *مزایا:*
    - سادگی: محاسبه ضرایب در این روش نسبت به اسپلاین مکعبی ساده‌تر است.
    - سرعت: این روش ممکن است در مواردی سریع‌تر از اسپلاین مکعبی باشد.
  - *معایب:*
    - کمترین دقت: در مقایسه با اسپلاین مکعبی، دقت تقریب در این روش کمتر است.
    - عدم پیوستگی: اسپلاین کوادراتیک نیازی به پیوستگی در نقاط انتهایی بازه‌ها ندارد.

- **اسپلاین نیوتنی:**
  - *مزایا:*
    - تعمیم‌پذیری: این روش می‌تواند برای تقریب توابع نیرو، حتی در صورت انحراف از صورت توابع معمولی، مؤثر باشد.
    - سادگی در افزایش درجه: افزایش درجه اسپلاین نیوتنی باعث افزایش دقت در نقاط تقسیم بازه‌ها می‌شود.
  - *معایب:*
    - کمترین دقت در موارد پیچیده: در توابع پیچیده با انحراف‌های زیاد از مدل نیوتنی، دقت این روش کاهش می‌یابد.
    - عدم پیوستگی: همچنان مشکلات عدم پیوستگی در نقاط انتهایی بازه‌ها ممکن است باقی بماند.

شرح کد و بررسی نتایج:

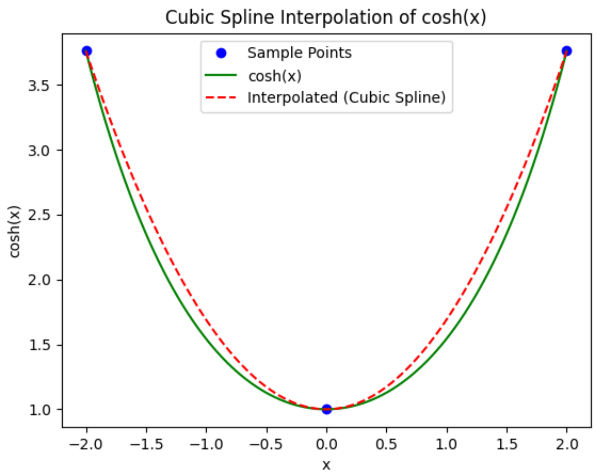
دیرکتوری کد ها به صورت زیر است:

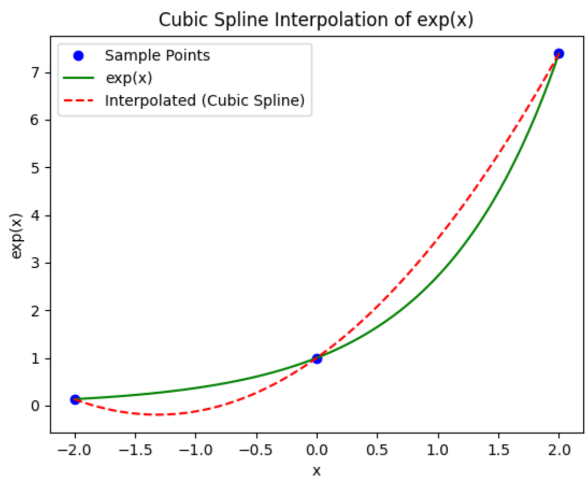
- compare_performance.py
- quadradic_interpol.py
- cubic_interpol.py
- newtonian_interpol.py
- polar_interpol.py
- test.py

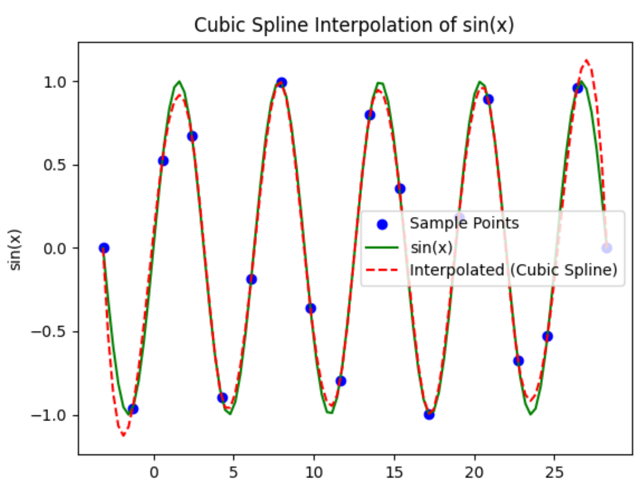
اسکریپت اول در دومرحله ابتدا از نظر زمانی و بعد از نظر دقت درونیابی و تقریب با توجه به تعداد داده های سمپل، سه روش درونیابی اسپلاین مکعبی و مربعی و درونیابی نیوتونی را با یکدیگر مقایسه می کند.

سه اسکریپت بعدی به ترتیب درونیابی اسپلاین مربعی،درونیاب اسپلاین کعبی و درونیابی نیوتونی برای هر سه تابع exponential, cosine hyperbolic , sine را با سمپل سایز ۱۸ برای سینوس در بازه [-pi,9pi] و سمپل سایز۳ برای کسینوس هایپربولیک و نمایی در بازه [-2,3] انجام میدهد و نمودار بدست آمده برای هر کدام مطابق با تصاویر زیر است:

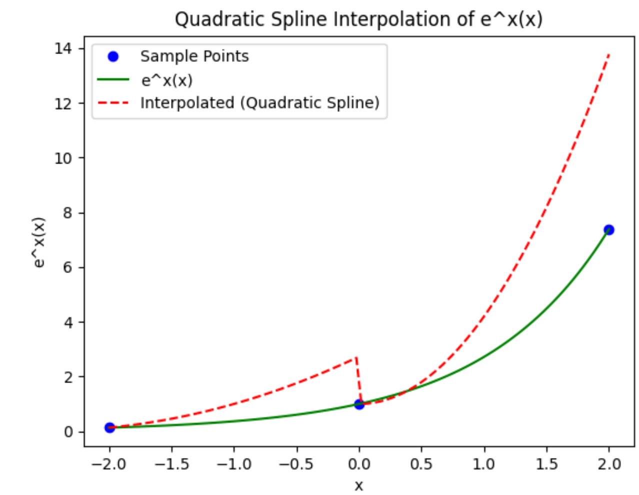
داده های درونیابی اسپلاین مکعبی:

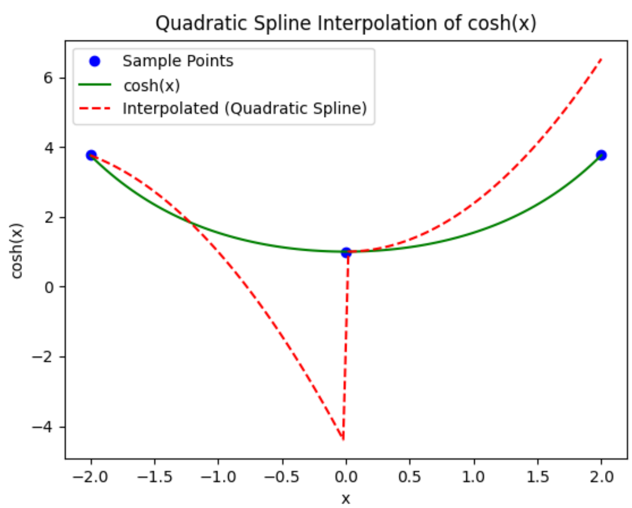


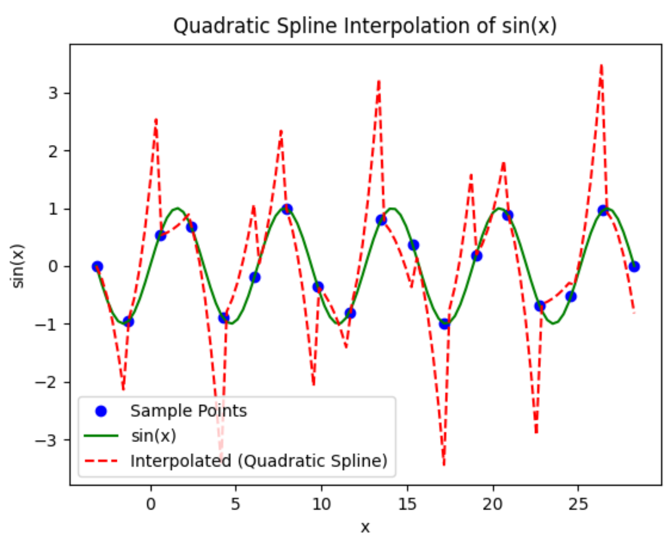




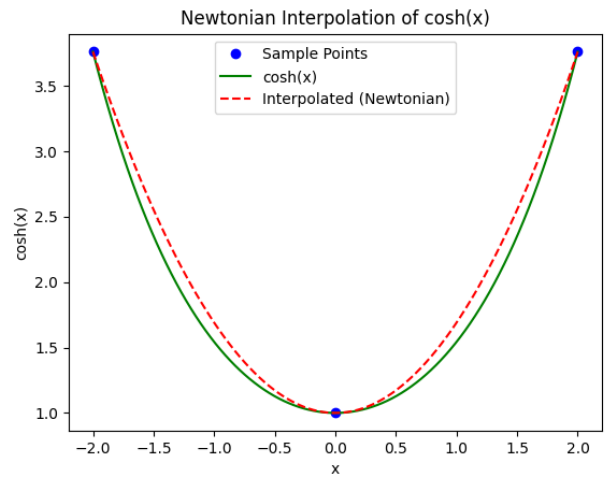
داده های درونیابی اسپلاین مربعی:

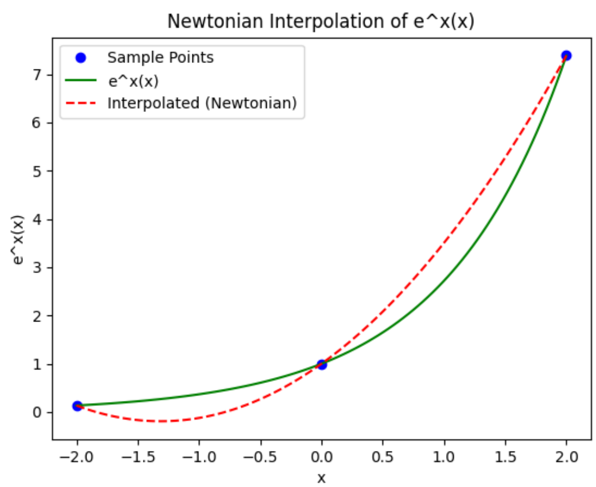


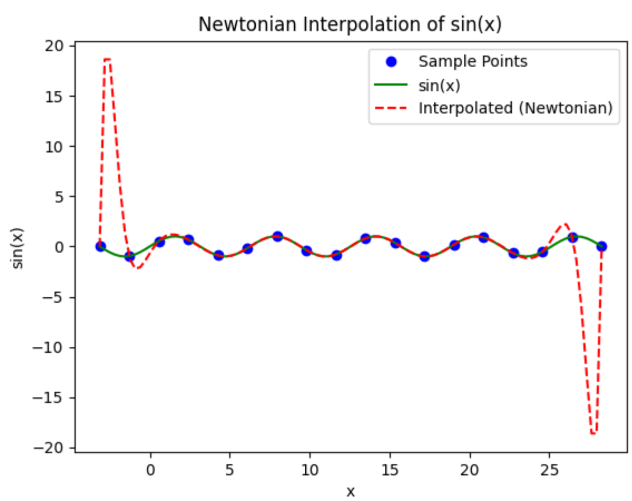




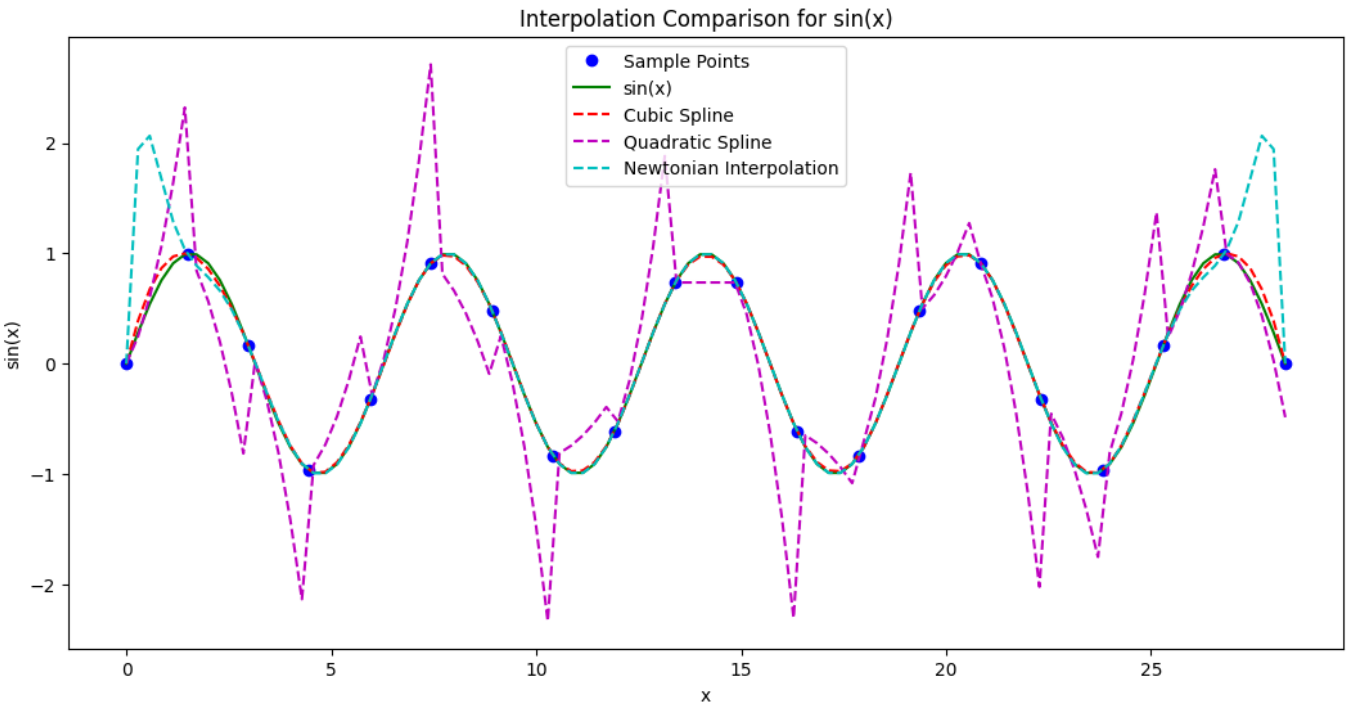
داده های درونیابی نیوتونی:

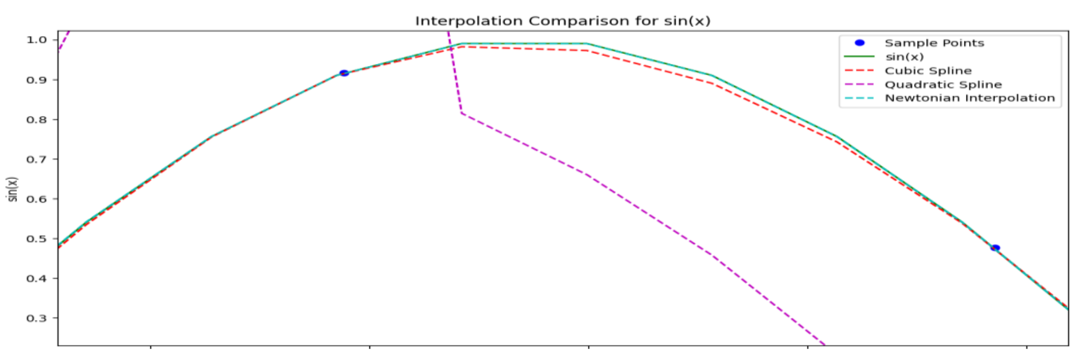

Newtonian Interpolation of cosh(x)
Sample Points
cosh(x)
Interpolated (Newtonian)
cosh(x)
x

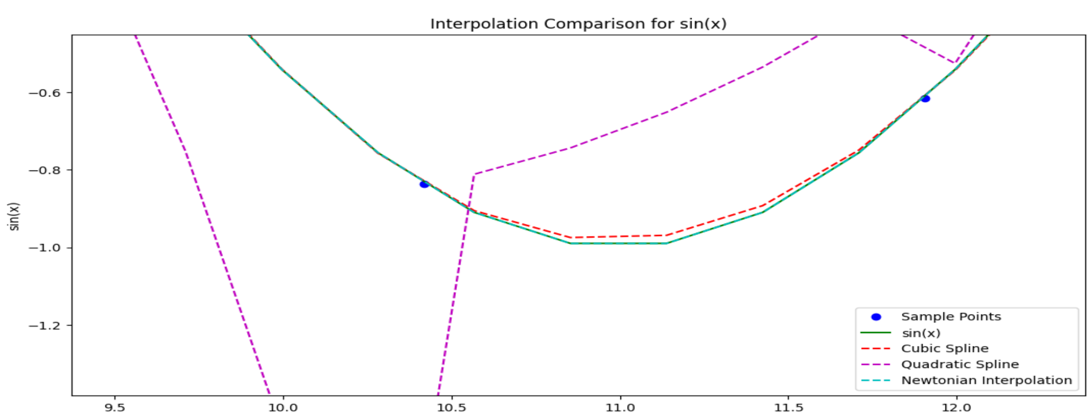

Newtonian Interpolation of e^x(x)
Sample Points
e^x(x)
Interpolated (Newtonian)
e^x(x)
x


Newtonian Interpolation of sin(x)
Sample Points
sin(x)
Interpolated (Newtonian)
sin(x)
x

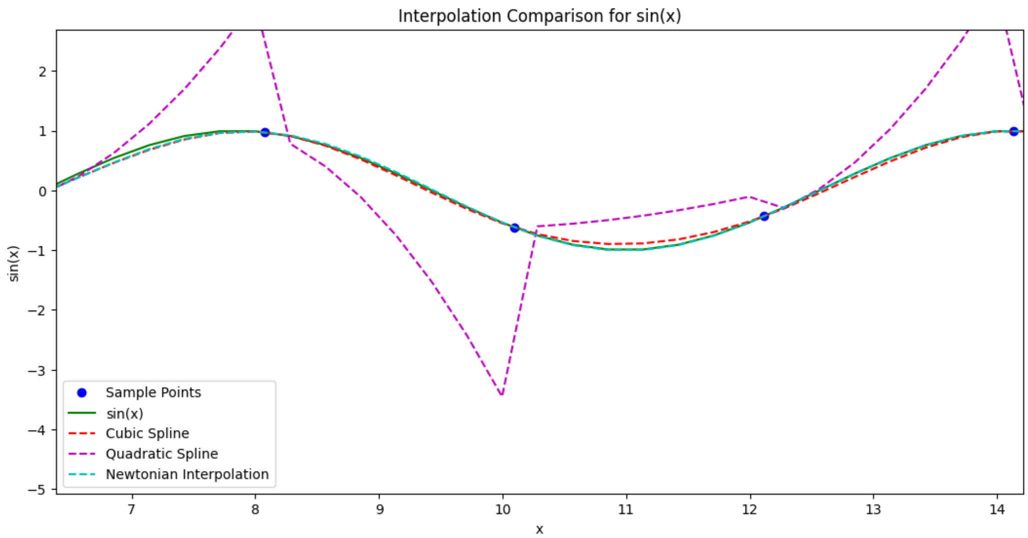
حال به بررسی خروجی اسکریپت مقایسه عملکردی این ۳ متد میپردازیم:







نمودار صفحه قبل به نمودار سینوس در بازه [-pi,9pi] است که هر سه روش درون یابی بر روی آن در کنار هم انجام و آورده شده است. با سمپل سایز ۲۰ نقطه و همانطوردر پلات های زوم شده در نقاط اکسترمم تابع مشهود است درونیابی نیوتونی درون بازه بسیار دقیق تر از دو متد دیگر است اما با این حال در بازه های ابتدایی و انتهایی حتی از درون یابی مربعی نیز پرت تر است.



نمودار بالا با ۱۵ سمپل دیتا رسم شده که دقت کمتری دارد و زوم شده د بازه درونی سمپل ست اس،آنچه مشهود است قت بالاتر درونیابی نیوتونی درون بازه نسبت به دو متد دیگر است

مقایسه عملکرد زمانی:

```
Cubic Spline Time: 0.0006767000304535031 seconds
Quadratic Spline Time: 6.190000056758523e-05 seconds
Newtonian Interpolation Time: 0.03800629999022931 seconds
```

از لحاظ عملکرد زمانی برای درونیابی با استفاده از ۲۰ سمپل درست است که درونیابی اسپلاین مربعی زمان کمتری طی کرده اما دقت آن به صرت قابل توجهی کمتر از دو متد دیگر است که زمان کمتر(از مرتبه 10^-5) که تنها در ۱۰۰میلی ثانیه با اسپلاین مکعبی فاصله دارد(۱۰/۱).

قسمت جالب توجه اما مدت زمان الگوریتم نیوتونی در مقایسه با اسپلاین مکعبی است که چیزی حدود ۵۰ برابر بیشتر زمان برده اما دقت آن به جز در نقاط اکسترمم(نقاطی که به صورت کلی در دیتا ست نقاطی که در بازه تحول رفتار مدل قراردارند) دقت قابل توجهی نسبت به درونیابی مکعبی ندارد و اگر سمپل سایز را افزایش دهیم این اختلاف دقت نیز از بین رفته اما همچنان بدلیل الگوریتم درونیابی نیوتونی که اساسا پیچیده تر از الگوریتم اسپلاین مکعبی (که صرفا بدست آورد ضرایب یه معادله درجه ۳ با توجه به سمپل ست است) هست،این تفاوت زمانی خواهد بود و حتی بیشتر نیز می شود.

نتیجه گیری :

در نهایت، انتخاب بهترین و روشن ترین روش بستگی به نیازهای خاص مسئله و ویژگی‌های داده‌های قرار به تقریب است. هر یک از روش‌های تقریب سرراست، رومباز، و اینترپولیشن نیوتونی با مزایا و محدودیت‌های خود همراه هستند.

- **اسپلاین مکعبی:**
  - **مزایا:** دقت بالا، نرمی و ادامه‌پذیری. مناسب برای انواع توابع پیوسته و صعودی.
  - **محدودیت‌ها:** پیچیدگی محاسبات و نگه‌داری ضرایب برای هر بازه، و ممکن است در توابع پیوسته و مرزهای بازه‌ها، افزون وزن‌دهی وجود داشته باشد.

- **اسپلاین درجه دوم:**
  - **مزایا:** سادگی محاسبه ضرایب و ممکن است محاسبات سریع‌تر باشد. مناسب برای مواردی که دقت بالا ضروری نیست.
  - **محدودیت‌ها:** دقت کمتر نسبت به اسپلاین مکعبی، عدم ادامه‌پذیری در مرزهای بازه.

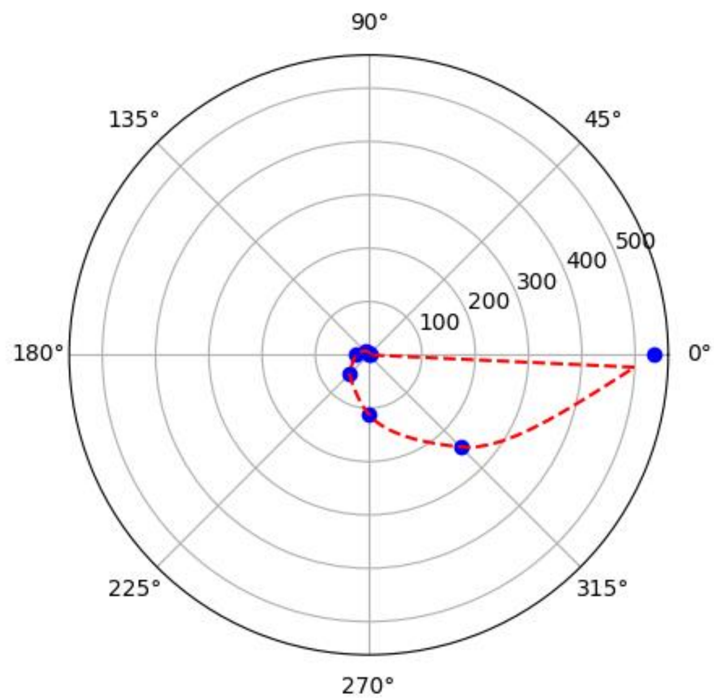
- **اسپلاین نیوتونی:**
  - **مزایا:** قابلیت تعمیم به راحتی به طیف وسیعی از توابع، امکان افزایش درجه برای دقت بیشتر در تقسیم بازه‌ها.
  - **محدودیت‌ها:** ممکن است در توابع پیچیده با انحراف‌های قابل توجه از مدل نیوتونی، دقت پایین‌تری داشته باشد و در مرزهای بازه ادامه‌پذیری نداشته باشد.

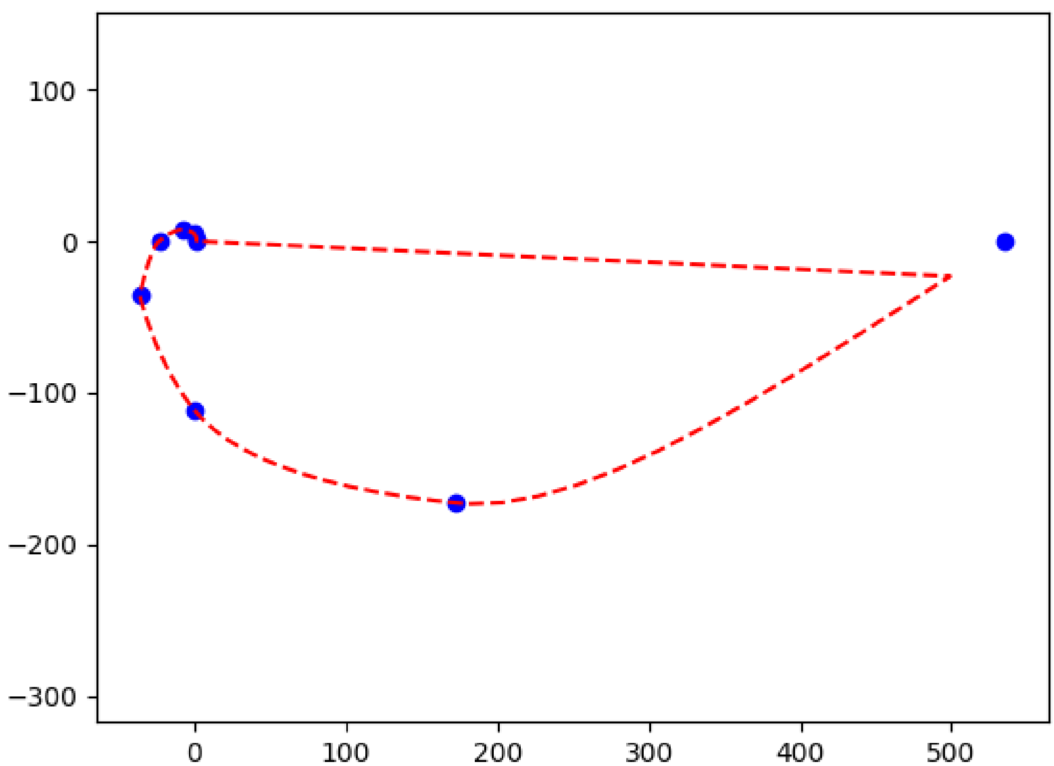
انتخاب بهترین روش یک موازنه بین دقت، کارایی محاسباتی و نرمی موردنیاز در تابع تقریبی را می‌طلبد. برای توابع با پیچیدگی بالا و رفتار نامنظم، قدرت اسپلاین مکعبی باعث می‌شود آنها گزینه‌ای ترجیحی باشند. در حالتی که سادگی و سرعت بالا بر دقت بیشتر حاکم است، استفاده از اسپلاین درجه دوم ممکن است مناسب‌تر باشد. اسپلاین نیوتونی یک رویکرد انعطاف‌پذیر است، به ویژه زمانی که با داده‌های نامنظم سروکار داریم، اما عملکرد آن به اندازه اعتقاد به مدل نیوتونی از انطباق داده به مدل وابسته است.

در نهایت، استفاده از یک روش خاص و یا ترکیب متدها باید با توجه به ویژگی‌های منحصر به فرد هر مسئله انجام شود.

ضمیمه

علاوه بر متد های بررسی شده،که همگی در دستگاه کارتزینی (دکارتی) تعریف شده بودند، درونیابی قطبی برای بررسی دیتای مختلط/قطبی کاربردی است،در ادامه سعی شده تا با استفاده از کتابخانه های پایتون تابع f=e^theta را با استفاده از متد درونیابی قطببی و سمپل سایز ۹، تقریب زده (در بازه [0,2pi] <- theta) و رسم نموده ،نمودار ثانی نیز تبیل نمودار اول در دستگاه کارتزینی است: (اسکریپ polar_interpol.py)


90°
135°
45°
180°
0°
225°
315°
270°
100
200
300
400
500


100
0
−100
−200
−300
0
100
200
300
400
500