Содержание отчета о выполнении задания по численным методам

1. В задании 2 необходимо построить полином Лагранжа для двух функций: f1 и f2 на отрезке от 0 до 2. Также, для вычислений (для второй функции), я использовал интерполяцию сплайнами, а также метод конечных разностей (для вычисления производной пятого порядка). Я ссылался на следующую литературу:

- А. А. Самарский - «Введение в теорию разностных схем.» «Наука», М., 1971 (метод конечных разностей)

- Д.П. Костомаров, А.П. Фаворский – «Вводные лекции по численным методам: Учебное пособие.» -М.: Логос, 2004. – 184 с.: ил.» (для построения интерполяционного полинома Лагранжа)

Задачи интерполяции часто могут возникать в реальной жизни, в разных сферах. Например, в анализе данных (например, для обучения ИИ). Для того чтобы «устранить пробелы» в наборе данных, можно создать дополнительные, которые нужны для обучения, за счет задачи интерполяции. Также пробелы могут возникать и во время проведения экспериментов, или метеорологических наблюдений. Это нужно для создания целостных данных.

Еще одним примером может быть использование интерполяции для улучшения качества изображения. Рассчитывая промежуточные данные, исходя из изображения, можно улучшить его качество за счет увеличения его разрешения. Это используется также и в 3д графике, чтобы улучшать качество текстур (например, в играх).

2. Рассмотрим более подробно каждый численный метод:

1) Построение интерполяционного полинома Лагранжа

Для решения задачи интерполяции мы строим полином Лагранжа, значения которого в узлах интерполяции (в нашем случае, в 3, 5, 9 и 17 точках) будут совпадать со значениями функции fi(x). Обозначим его как Pn(x). В свою очередь, для построения полинома нам необходимо построить базисные полиномы, которые будут «ориентированы» в узлах сетки. Формула такого полинома: произведение П при j от 0 до n (j != i): (x – xi) / (xi – xj). Этот полином принимает значение 0 при x = xj (j = i) и значение 1 при j != i. Обозначим такой полином как Qn,i(x). Осталось составить сумму для построения полинома Лагранжа: сумма по i от 0 до n f(xi) \* Qn,i(x). Это и будет являться формулой полинома Лагранжа. Она реализована программно в моем коде, в функции Lagrange\_polynomial(args). Функция вычисляет значение полинома в точке x.

Преимущества данного численного метода:

- Простота организации вычислительного процесса. Прост для понимания.

- При небольшом количестве узлов метод достаточно точен.

Недостатки:

- Если количество узлов слишком большое, то и нарушается точность вычислений. Возникает феномен Рунге для полиномов больших степеней.

- Если понадобится добавить новый узел, то полином нужно будет строить заново.

- При больших осцилляциях (колебаниях) функции (как на 2 наборе данных), возникают также слишком большие отклонения в вычислениях (особенно ближе к границам интервала).

Поэтому, для второго набора данных, я использовал метод кубических сплайнов.

2) Интерполяция сплайнами

В этом методе используется кусочно-полиномиальная интерполяция, которая оказывается эффективнее, чем интерполяция полиномом Лагранжа, так как функция разбивается на несколько полиномов третьей степени (что является относительно невысокой степенью), вместо вычисления полинома более высокого порядка. При этом, такое разбиение всё равно сохраняет точность.

Плюсами такого метода являются устойчивость к феномену Рунге, а также – что строится локальная интерполяция. То есть, если изменить один узел сетки, остальные узлы не меняются. Это также может упростить вычисления, ведь, как в построении полинома Ньютона, не придется строить каждый раз полином заново. Если удастся найти нужную интерполяцию на некоторых точках, то пересчет не потребуется.

Суть метода

Пусть у нас дан отрезок [a, b]. Строится его разбиение (сетка интерполяции) Искомый интерполянт S(x) должен обладать следующими свойствами:

1) На каждом сегменте [xi-1, xi] функция S(x) является полиномом третьей степени;

2) Сама функция и её первая S’(x) и вторая S’’(x) производные непрерывны на [a, b];

3) S(xi) = f(xi) = fi, i = 0, ... n;

4) S’’(a) = S’’(b) = 0.

Для отыскания полинома нам достаточно определить его коэффициенты на каждом из отрезков разбиения [a, b]. Соответственно, в моей программе для это была создана структура, содержащая в себе данные об этих коэффициентах. А сама функция, вычисляющая полиномы, просто сохраняет данные в массиве из этих структур.

2.3 Метод конечных разностей

Возникает вопрос, как можно посчитать производную пятого порядка в первом наборе данных (а также как посчитать производные для кубических сплайнов). Одним из самых простых способов является применение метода конечных разностей. Суть в том, чтобы разностно аппроксимировать производную функции некоторой сеточной функцией, которая в определенном наборе точек будет принимать приближенные значения производной. Соответственно, чем более высокого порядка будет производная функции, тем большее количество узлов сетки понадобится. Для начала, можем рассмотреть метод конечных разностей для аппроксимации производной первого порядка.

Мы можем в качестве аппроксимации взять правую (или левую) разностную производную в точке х (получаются исходя из определения производной). А также мы можем записать их линейную комбинацию (одну из) и получить формулу центральной разностной производной: v’(x) = (v(x + h) – v(x – h)) / 2h, где h > 0 и v(x) – сама функция, для которой записывается аппроксимация её производной. Заметим, что мы использовали 2 точки для сеточной функции: (x – h, x + h).

Далее, мы можем записать формулу аппроксимации второй производной:

V’’(x) = v(x + h) – 2v(x) + v(x – h) / h^2. Здесь использовалось 3 точки: (x – h, x, x + h).

Аппроксимировать мы можем производную любого порядка, вопрос лишь в том, насколько точной будет такая аппроксимация для более высоких порядков.

Тем не менее, я использовал этот метод для производной пятого порядка и получил следующую формулу:

V(5)(x) = (v(x – 3h) – 5v(x – 2h) + 10v(x – h) – 10v(x + h) + 5v(x + 2h) – v(x + 3h)) / h^5.

Точность метода: o(h^2)

3. Сходимость интерполяции

Для сходимости интерполяции, нам нужно найти сходимость функциональной последовательности: lim (n -> inf) max (x in [a, b])|Pn(x) – f(x)|. Соответственно, для сходимости этот предел должен равняться нулю.

Пока что, поскольку число узлов достаточно большое, есть вариант брать промежуточные точки (не равные xi, так как в них полином и функция совпадают), например, середины отрезов [xi-1, xi], и считать в них отклонения функций. В цикле находить максимальное отклонение. Затем увеличивать число узлов сетки, например, в 2 раза и снова считать отклонение. Повторить несколько раз, и найти, как изменяется максимум. Построить последовательность, и найти её предел. Если стремится к нулю, то интерполяционный процесс сходится.

4. Код программы

Моя программа создает несколько текстовых файлов, в каждый из которых записывает вычисления для каждой из функций и её интерполянтов. В файле сохраняется набор точек (аргументов функций и их значений), в дальнейшем можно выполнить визуализацию данных, например, в gnuplot.