МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет математики и компьютерных наук**

**Кафедра математических и компьютерных методов**

**Отчёт   
о прохождении  
производственной практики  
(педагогической практики)**  
**Б2.О.02.02(П)**

студент Пасько Дмитрий Анатольевич группа 202/1

Направление подготовки 02.04.01 Математика и компьютерные науки

Программа магистратуры «Математическое и компьютерное   
 моделирование»

Форма обучения очная

Квалификация выпускника магистр

Краснодар

2020

**СОДЕРЖАНИЕ**

Введение

О занятиях

О задаче

Формулы вейвлет-преобразования

Исходные коды

Результаты

Заключение

# ВВЕДЕНИЕ

В настоящее время преобладает много новых типов высших учебных заведений. Однако преобразования коснулись лишь той части педагогического процесса, которая включает в себя изменение технологии педагогического взаимодействия.

Переход к новой педагогике означает обновление всех сторон учебно-воспитательного процесса – его содержания, форм, методов, а что самое сложное – образа их мыслей, интересов и установок, их отношений друг к другу.

Следовательно, объём знаний, необходимых современному специалисту, возрастает, но срок обучения в ВУЗе ограничен несколькими годами. Отсюда следует, что повышать свою квалификацию, обновлять свои знания любому специалисту придётся всю жизнь. В студенческие годы он обязан освоить с азов методику самообразования, научится критически мыслить и находить новые способы решения профессиональных задач.

Без научного подхода к решению вопроса о направленности, содержании и методике воспитания и обучения студентов ВУЗов невозможно обеспечить эффективность подготовки специалистов.

В настоящее время учебный процесс в ВУЗах стал более сложным по своим задачам, интенсивности и содержанию. Он требует глубокого практического осмысления преподавателями закономерностей учебной деятельности, принципов и методов обучения и воспитания, формирования личности.

Всё более очевидным становится мнение, что без использования практических педагогических знаний нельзя выработать всестороннюю подготовленность студентов к успешной профессиональной деятельности, обеспечить высокий педагогический уровень их обучения и воспитания, единства теоретической и практической подготовки с учётом профиля ВУЗа и специализации выпускников.

Подготовка будущего магистра осуществляется в процессе всей учебно-воспитательной работы в магистратуре университета, и важное место в этом занимает педагогическая практика. Она способствует воспитанию профессиональных интересов магистрантов, формированию личности будущего магистра, помогает получить первый опыт самостоятельной преподавательской работы, проверить на деле свои знания и способности, укрепить интерес к будущей профессии.

Педагогическая практика имеет целью создание условий, позволяющих магистрантам приобрести практические навыки самостоятельной педагогической деятельности, овладеть основами педагогического мастерства.

Требования к профессиональной подготовленности студента определяют следующие задачи практики:

* практическое ознакомление студентов с авторской методикой преподавания конкретного курса, обязательно входящего в базисный учебный план вуза;
* изучение учебно-методической литературы, лабораторного и программного обеспечения по рекомендованным дисциплинам учебного плана;
* разработка дополнительных методических и тестовых материалов для студентов в помощь преподавателю при ведении лекционных и семинарских занятий по курсу;
* получение представления о современных образовательных информационных технологиях;
* формирование и решение задач, возникающих в ходе педагогической деятельности и требующих углубленных профессиональных знаний;
* проведение педагогической работы с привлечением современных технологий;
* проведение анализа результатов работы;

разработка научно-методических материалов по темам учебных дисциплин.

# О занятиях

В ходе педагогической практики мною было проведено со студентами открытое практическое занятие по курсу «Эффективные вычисления в приложениях практической математики». Практическое занятие проходило в форме подготовки и проведения лекционного занятия.

Подготовка к занятию заключалась в детальном изучении материала по теме, составлении развернутого конспекта, который основывался на материалах методических указаний, учебников. Был подобран графический материал, иллюстрирующий основные положения темы, важные для понимания всего материала в целом. Для удобства работы материал был размножен и роздан студентам. Тема, цели и задачи занятия предварительно согласовывались с лектором данной дисциплины.

# О задаче

В процессе преподавания предмета я тесно обращался к нескольким практическим примерам, с которыми сталкивался в процессе своей научной деятельности. Основная из них – преобразование кода на устаревшем языке в современный и эффективный код, который в будущем намного проще поддерживать. Этот аспект был продемонстрирован на задаче Габора.

Поставленная задача заключалась в переводе скриптов вейвлет-преобразования Габора с исходного Matlab на Python и ускорении полученного кода.

Лаборатория ИММИ по большей части занимается задачами поиска дефектов (царапин, трещин, коррозий) в волноводах (металлических и других пластинах). Одно из решений этой задачи, с помощью построения эллипсов (с реализациями этого и других решений можно ознакомиться в репозитории <https://github.com/PasaOpasen/Search-for-defects-in-plates>), на одном из этапов требует знания параметров , дающих максимальное значение функции вейвлет-преобразования исходных сигналов, полученных с датчиков на волноводе. Поскольку для поиска максимума этой весьма сложной функции требуется знание многих её значений (речь идёт о десятках тысяч), очень важно максимально ускорить выполнение операций получения значений функции.

Так как Matlab сам по себе является очень медленным и трудноподдерживаемым языком (в плане обновления библиотек и разных вспомогательных утилит), многие коды желательно переписать на Python, являющийся одним и самых популярных и поддерживаемых языков с множеством удобный библиотек для научной деятельности.

Чтобы добиться поставленных целей, я изучил и посоветовал изучить студентам следующие руководства:

* Svein Linge. Hans Petter Langtangen. Programming for Computations – Python
* Python High Performance by Gabriele Lanaro (<https://github.com/PasaOpasen/ProgrammingBooks/blob/master/Python/Python%20High%20Performance%20by%20Gabriele%20Lanaro%20(z-lib.org).pdf>)
* Robert Johansson. Numerical Python (<https://github.com/PasaOpasen/ProgrammingBooks/blob/master/Python/%5BRobert-Johansson%5D-Numerical-Python_-Scientific-Co(z-lib.org).pdf>)

# Формулы вейвлет-преобразования

Пусть – сигнал, определённый для каких-то времён . Тогда прямое вейвлет-преобразование этого сигнала имеет вид:

где – сам вейвлет, черта обозначает комплексное сопряжение. Часто используется обозначение

Для вейвлета Габора эта функция выглядит так:

где – константы, – коэффициент Габора.

Обратное вейвлет-преобразование имеет вид:

где – фурье-преобразование вейвлета.

# Исходные коды

Исходные файлы Matlab расположены по адресу <https://github.com/PasaOpasen/Gabor-wavelet/tree/master/matlab_source>. Все результаты и некоторое описание находятся в репозитории <https://github.com/PasaOpasen/Gabor-wavelet>.

# Результаты

После преподавания курса в тесной связи с конкретной задачей я предложил студентам попробовать добиться ещё лучших результатов. У нескольких человек это получилось.

Содержание курса заключено в следующем. Перед созданием обратного преобразования нужно найти оптимальную конфигурацию, дающую прямое преобразование. Поскольку язык python является интерпретируемым, сам по себе он не быстрый, однако для него создана масса библиотек, оптимизированных под научные вычисления (в том числе numpy) и обращающихся к скомпилированным функциям, написанным на C или Fortran. Кроме того, имеется множество способ кеширования и параллелизации кода, а также несколько разных парадигм jit-компиляции. Чередуя техники оптимизации вычислений с указанными вариантами ускорения кода, можно добиться невероятных результатов. По итогу работы удалось ускорить первоначальный вариант кода в 200 раз, не используя параллелизма, и более чем в 800 раз, используя его.

Взглянем на самый прямой перевод исходного кода на python из файла <https://github.com/PasaOpasen/Gabor-wavelet/blob/master/translation_only.py>:

def DWT\_signal(ut, a, b, t0, AA, BB, TT, omega0, Gabor\_coef):

h\_step=t0[1]-t0[0];

psi\_t = np.empty(TT, dtype = np.complex128)

Wab = np.empty((BB, AA), dtype = np.complex128)

for j in range(AA):

for i in range(BB):

for k in range(TT):

t\_cur=(t0[k]-b[i])/a[j]

psi\_t[k] = GaborWavelet(omega0, t\_cur, Gabor\_coef).conjugate()

f\_psi = psi\_t \* ut

Wab[i,j] = 0.5\*(f\_psi[0]+f\_psi[-1])

for k in range(2, TT-1):

Wab[i,j] = Wab[i,j] + f\_psi[k]

Wab[:,j] = Wab[:,j]\*h\_step/math.sqrt(a[j])

return Wab

Его главная проблема – это суммирование в цикле, ведущее к постоянным обращениям к одним и тем же элементам массива, хотя нет необходимости менять значения этих элементов.

Векторизовав некоторые участки указанного кода, получаем следующую функцию (<https://github.com/PasaOpasen/Gabor-wavelet/blob/master/light_vectorization.py>):

def DWT\_signal(ut, a, b, t0, AA, BB, TT, omega0, Gabor\_coef):

h\_step = t0[1]-t0[0]

Wab = np.empty((BB, AA), dtype = np.complex128)

for j in range(AA):

for i in range(BB):

t\_cur=(t0-b[i])/a[j]

psi\_t = GaborWavelet(omega0, t\_cur, Gabor\_coef).conjugate()

f\_psi = psi\_t \* ut

Wab[i,j] = 0.5\*(f\_psi[0]+f\_psi[-1]) + np.sum(f\_psi[2:TT-1])

Wab[:,j] \*= h\_step/math.sqrt(a[j])

return Wab

Можно пойти чуть дальше и векторизовать операции над столбцами матрицы в операции над всей матрицей сразу (<https://github.com/PasaOpasen/Gabor-wavelet/blob/master/strong_vectorization.py>):

def DWT\_signal(ut, a, b, t0, AA, BB, TT, omega0, Gabor\_coef):

h = (t0[1]-t0[0])

Wab = np.empty((BB, AA), dtype = np.complex128)

for j in range(AA):

for i in range(BB):

t\_cur=(t0-b[i])/a[j]

psi\_t = GaborWavelet(omega0, t\_cur, Gabor\_coef).conjugate()

f\_psi = psi\_t \* ut

Wab[i,j] = 0.5\*(f\_psi[0]+f\_psi[-1]) + np.sum(f\_psi[2:TT-1])

Wab = np.multiply(Wab, h/np.sqrt(a))

return Wab

Это максимально векторизованный код, который можно получить. Он работает в 20 раз быстрее первой версии. Можно попробовать ускорить код ещё больше, используя jit-компиляцию. Оптимизируем первую версию кода в плане индексов и просто применим jit-компиляцию (<https://github.com/PasaOpasen/Gabor-wavelet/blob/master/numba_just.py>):

@jit()

def DWT\_signal(ut, a, b, t0, AA, BB, TT, omega0, Gabor\_coef):

h\_step=t0[1]-t0[0];

psi\_t = np.empty(TT, dtype = np.complex128)

Wab = np.empty((BB, AA), dtype = np.complex128)

for j in range(AA):

for i in range(BB):

for k in range(TT):

t\_cur=(t0[k]-b[i])/a[j]

psi\_t[k] = GaborWavelet(omega0, t\_cur, Gabor\_coef).conjugate()

f\_psi = psi\_t \* ut

s = 0.5\*(f\_psi[0]+f\_psi[-1])

for k in range(2, TT-1):

s+= f\_psi[k]

Wab[i,j] = s

Wab[:,j] = Wab[:,j]\*h\_step/math.sqrt(a[j])

return Wab

Этот вариант работает уже в 200 раз быстрее самого первого. Можно ускорить его на 3-5%, добавив векторизацию (<https://github.com/PasaOpasen/Gabor-wavelet/blob/master/numba_vectorization.py>):

@jit()

def DWT\_signal(ut, a, b, t0, AA, BB, TT, omega0, Gabor\_coef):

h\_step=t0[1]-t0[0]

psi\_t = np.empty(TT, dtype = np.complex128)

Wab = np.empty((BB, AA), dtype = np.complex128)

for j in range(AA):

for i in range(BB):

for k in range(TT):

t\_cur=(t0[k]-b[i])/a[j]

psi\_t[k] = GaborWavelet(omega0, t\_cur, Gabor\_coef).conjugate()

f\_psi = psi\_t \* ut

Wab[i,j] = 0.5\*(f\_psi[0]+f\_psi[-1]) + np.sum(f\_psi[2:TT-1])

Wab = np.multiply(Wab, h\_step/np.sqrt(a))

return Wab

Этот вариант лишь немного быстрее, зато дисперсия времён, затраченных на его выполнение, в два раза меньше, то есть этот вариант более стабилен.

Наконец, используем параллелизацию по внешнему (самому первому) циклу (это наиболее правильный вариант <https://github.com/PasaOpasen/Gabor-wavelet/blob/master/numba_vec_parallel.py>):

@jit(parallel = True)

def DWT\_signal(ut, a, b, t0, AA, BB, TT, omega0, Gabor\_coef):

h\_step = t0[1]-t0[0]

psi\_t = np.empty(TT, dtype = np.complex128)

Wab = np.empty((BB, AA), dtype = np.complex128)

for j in prange(AA):

for i in range(BB):

for k in range(TT):

t\_cur=(t0[k]-b[i])/a[j]

psi\_t[k] = GaborWavelet(omega0, t\_cur, Gabor\_coef).conjugate()

f\_psi = psi\_t \* ut

Wab[i,j] = 0.5\*(f\_psi[0]+f\_psi[-1]) + np.sum(f\_psi[2:TT-1])

Wab = np.multiply(Wab, h\_step/np.sqrt(a))

return Wab

Этот вариант быстрее предыдущего примерно в 2/3 \* число процессоров раз.

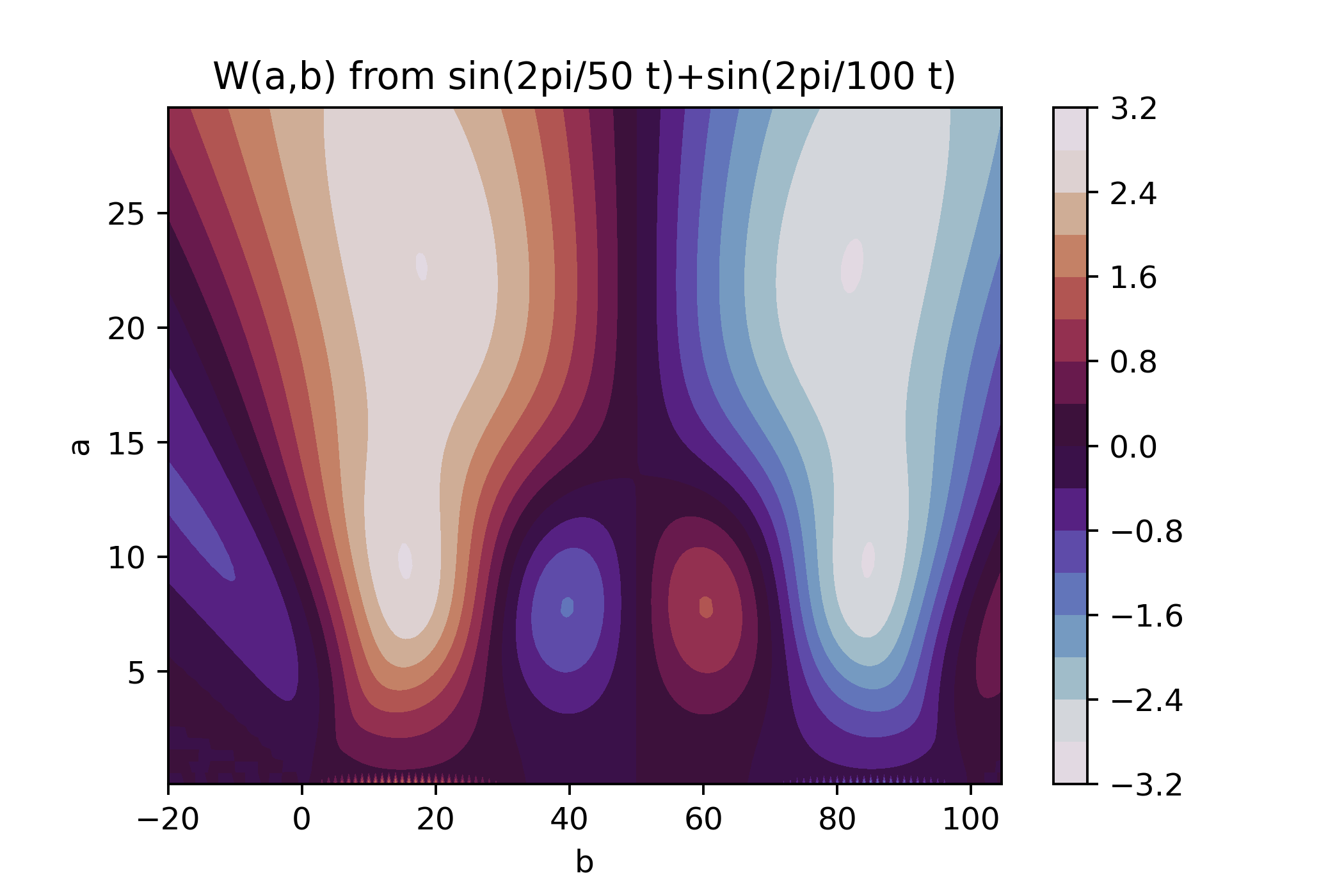
Подытоживая результаты, приведём таблицу бенчмарков:

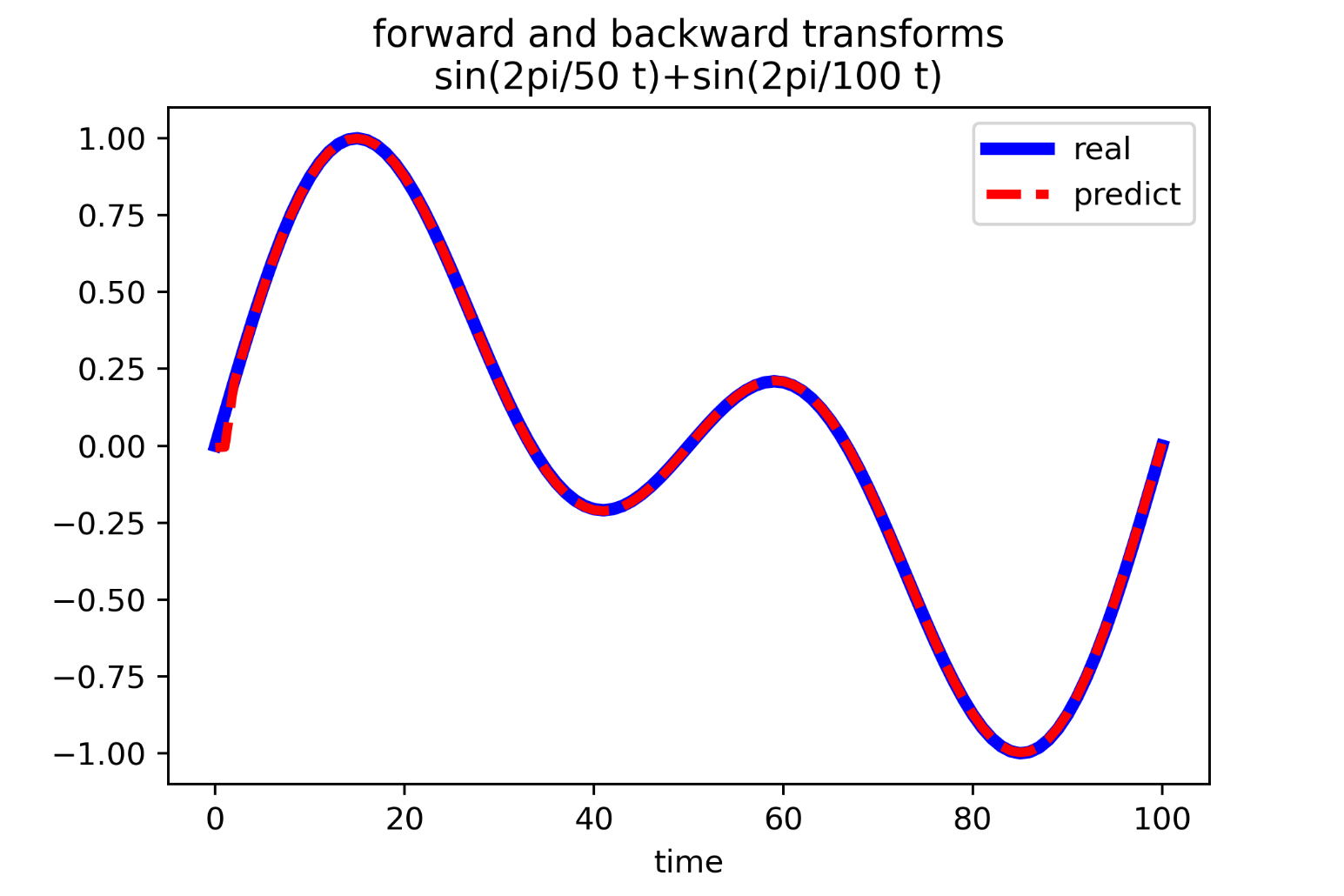
| **Version** | **Time** |
| --- | --- |
| translation\_only | 7.24 s ± 142 ms per loop (mean ± std. dev. of 7 runs, 1 loop each) |
| light\_vectorization | 372 ms ± 8.83 ms per loop (mean ± std. dev. of 7 runs, 1 loop each) |
| strong\_vectorization | 368 ms ± 8.54 ms per loop (mean ± std. dev. of 7 runs, 1 loop each) |
| numba\_strong | 219 ms ± 5.17 ms per loop (mean ± std. dev. of 7 runs, 1 loop each) |
| numba\_just | 36.3 ms ± 914 µs per loop (mean ± std. dev. of 7 runs, 10 loops each) |
| numba\_vectorization | 35.1 ms ± 350 µs per loop (mean ± std. dev. of 7 runs, 1 loop each) |
| numba\_vec\_parallel | 8.33 ms ± 85.2 µs per loop (mean ± std. dev. of 7 runs, 100 loops each) |

Итоговый результат лучше исходного в 800 раз!

Используя наилучшим образом оптимизированное прямое вейвлет-преобразование, было реализовано и обратное преобразование.

Визуализации этих преобразований для функции показаны на следующих рисунках:





# ЗАКЛЮЧЕНИЕ

В ходе прохождения педагогической практики мною выполнены все пункты педагогического задания.

При знакомстве с работой кафедры и при подготовке к практическим и лекционным занятиям понравилось отношение преподавателей ко мне. Они всегда старались прийти на помощь, дать ответ на поставленные вопросы.

Оставляет лучшее сама методика преподавания и то обстоятельство, что конспект лекций официально не требуется, но при подготовке новых курсов каждый преподаватель пишет его для себя, так как с ним работать на занятиях легче. Но дальше конспект начинает мешать, так как курсы постоянно требуют обновления, и хороший конспект превращается у преподавателя в набор почерканных, переклеенных листков. Это стимулирует переход с бумажного делооборота в электронный.

При проведении учебных занятий понравилось отношение студентов ко мне, даже на старших курсах было деловые взаимоотношения на лекционных занятиях. С непривычки сложно было в начале лекции что-то рассказывать, после стало получаться лучше.

При проведении практической работы по осуществлению расчетов в ходе выполнения задания, была осуществлена коррекция режима в рассчитанной схеме и выбор необходимых технических решений с точки зрения эффективности.

Считаю, что пройденная мной педагогическая практика является не просто необходимой, но и обязательной для дальнейшей моей деятельности.