دانشکده علوم ریاضی
دانشگاه صنعتی شریف

تحقیق در عملیات ۱

محمدهادی فروغمنداعرابی
پاییز ۱۳۹۹

حل دستگاه تنک ۲

جلسه نوزدهم

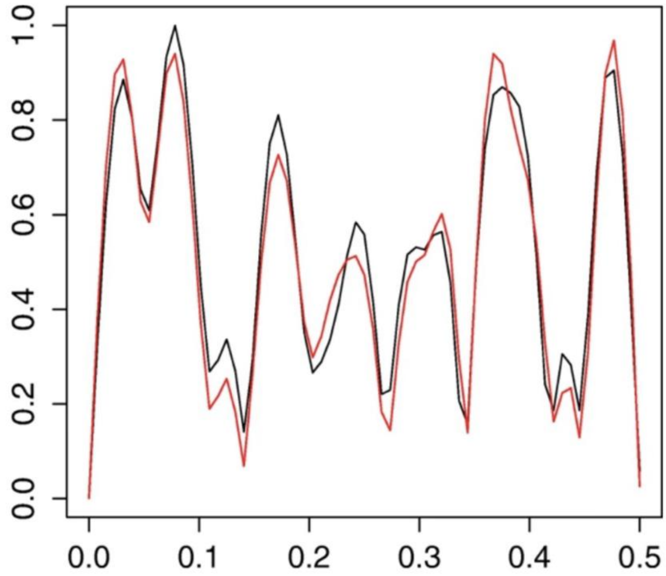
نگارنده: آریا رضائیان

۱ تبدیل فوریه

ایده اولیه شکل گیری تبدیل فوریه:
چه شکل هایی را می توان با توابع سینوسی ساخت ؟

به وضوح می توان نتیجه گرفت همه شکل ها را نمی توان با توابع سینوسی ساخت چون در مضارب عدد پی این توابع صفر می شوند.

$$f(x) = \sum_{n=۱}^{\infty} a_n \sin nx$$

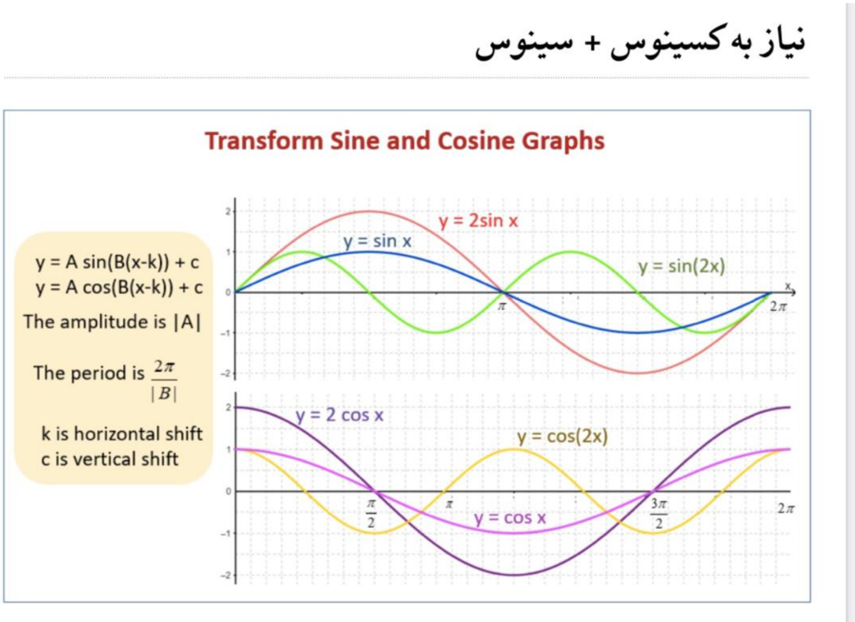


شکل ۱

حال می خواهیم این سوال را بررسی می کنم که با جمع کردن تعدادی توابع سینوسی و کسینوسی چه توابعی را می توان ساخت ؟

$$f(x) = \sum_{n=1}^{\infty} a_n \sin nx + b_n \cos nx$$

پاسخ فوریه به این سوال این بود که تقریبا همه توابع را می توان با توابع سینوسی و کسینوسی ساخت.
یعنی فرکانس ها پایه ای هستند برای تقریبا تمامی توابع.



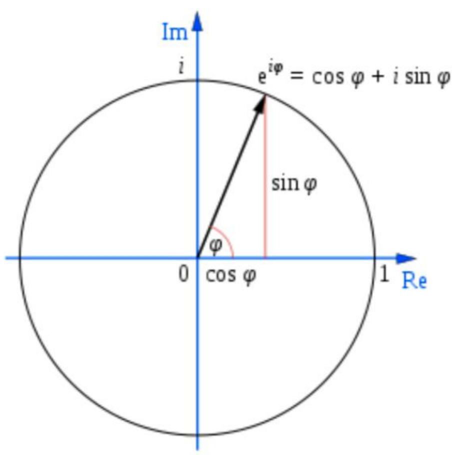
شکل ۲

در واقع تبدیل پایه توابع به توابع فرکانس را تبدیل فوریه می گویند.

اما نوشتن توابع به صورت جمع تعدادی توابع سینوسی و کسینوسی مطلوب ما نیست. بنابراین سعی می کنیم همه توابع را به صورت جمع توابع مختلط که شامل سینوس و کسینوس هستند، بنویسیم.

$$f(x) = \sum_{n=۰}^{\infty} a_n e^{\frac{-inx}{۲\pi}}$$

در نتیجه هر نقطه از این تابع ، یک نقطه روی دایره مثلثاتی است که به صورت یک ترکیب خطی از توابع سینوسی و کسینوسی است.



شکل ۳

اما هدف ما در این قسمت بررسی توابع گسسته است که لزوما روی R تعریف نشده اند.
به عنوان مثال می خواهیم توابع به شکل

$$a = (a_0, a_1, a_2, a_3, ..., a_{n-1})$$

را به توابعی مانند

$$\hat{a} = (\hat{a_0}, \hat{a_1}, \hat{a_2}, \hat{a_3}, ..., \hat{a}_{n-1})$$

ببریم.
که در آن:

$$\hat{a_u} = \frac{1}{n}\sum_{j=0}^{n-1} a_j w^{-uj}$$

به طور خلاصه می توان تبدیل فوریه را این گونه تعریف کرد که اگر بردار a را داشته باشیم ،به وسیله ضرب کردن بردار F در آن ، بردار جدید با پایه های فرکانسی تولید می شود، که در آن:

$$F_{uj} = \frac{1}{n} w^{-uj}$$

۲ کاربردهای تبدیل فوریه

۱.۲ خروجی های تنک

یکی از خاصیت های تبدیل فوریه این است که اگر یک بردار با تعداد زیادی مولفه به آن بدهیم، با نگه داشتن تعداد کمی از آن مولفه ها، خاصیت آن بردار را حفظ می کند.

۲.۲ محاسبه سریع DFT

در قسمت قبل می دانیم روش بدیهی ضرب کردن بردار F در بردار a از مرتبه $O(n^2)$ زمان می گیرد. حالی می توانیم با استفاده از الگوریتم FFT زمان را به O(nlogn) کاهش دهیم.

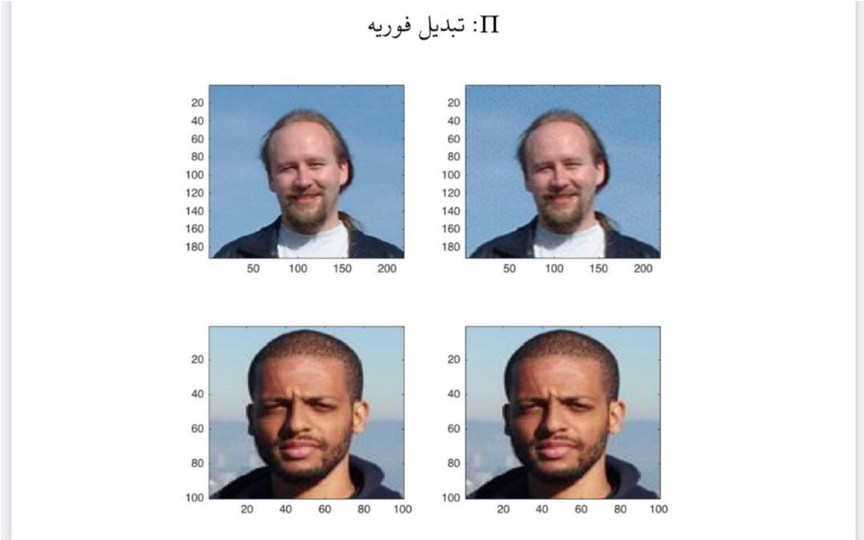
۳.۲ فشرده سازی تصویر

فرض کنید بردار u به طول n که در آن طول n بسیارزیاد است را داریم. می خواهیم با یک تبدیل آن را به بردار Su تبدیل کنیم که در آن Su برداری بزرگ اما تنک است.

حال اگر می دانستیم قسمت های غیر تنک این بردار کجا واقع شده اند، می توانستیم با ضرب کردن بردار A در آن، صرفا بردار ASu را با طول k ذخیره کنیم(k تعداد مولفه های غیرصفر است)
که در آن k به مراتب از n کوچکتر است .

حال برای بازیابی تصویر نیز می توانستیم معکوس تبدیل فوریه را اعمال کنیم و به تعداد n-k به انتهای بردار ASu صفر اضافه کنیم تا تصویر اصلی بازیابی شود.

در نتیجه الگوریتم بازیابی به صورت زیر در می آید:
۱ـ بردار q را ازافزودن n-k صفر به ASu بدست می آوریم.
۲ ـ جواب: با ضرب کردن معکوس S در q بردار اولیه بازیابی می شود.



شکل ۴

۳ ارجاع و منابع

Matousek ، Understanding and using linear programming