«به نام پروردگار»

گزارش پروژه اول درس جبرخطی

استاد درس: دکتر ناظرفرد

دانشجو: محسن محمدیان

شماره دانشجویی: ۹۸۳۱۵۰۲

باتشکر از گروه تدریس یاران

در این پروژه که به زبان پایتون هست؛ حل دستگاه معادلات به شکل ماتریس و با روش پله ای (echelon form) پیاده سازی شده است.برای این کار با استفاده از کتابخانه ی numpy و تابع array()، ابتدا ما یک ماتریس با درایه های صفر را که در کد پروژه با متغیر "matrix" تعریف شده می سازیم و سپس مقادیر ورودی را سطر به سطر از کاربر به صورت string دریافت کرده و با کمک تابع split در لیست "tmp " می ریزیم و در نهایت لیست tmp را با سطر مربوطه در "matrix" جمع می کنیم.(گرفتن سطرهای ماتریس در تابع gets_input() انجام می شود)

پس از آنکه ماتریس خود را initialize کردیم و کامل سطرهای ماتریس را از کاربر گرفتیم، مقادیر ثابت (مقادیر سمت راست های معادلات خطی) را از کاربر میگیریم و در "const_val" ذخیره می کنیم. سپس بااستفاده از تابع hstack()، constant valueها را به عنوان ستون آخر به ماتریس اضافه می کنیم تا augmented matrix ما بدست آید.

سپس باید سطرهای تمام صفر ماتریس (سطرهای صفر) با ردیف ها یا سطرهای آخر جابجا شوند، که این کار با تابع zeros_bottom() انجام پذیر است.

حال باید ماتریس را به فرم echelon در بیاوریم؛ که این کار با تابع make_echelonform() انجام می شود. در این تابع ما در دو حلقه ی موجود در تابع تکرار می شویم و هر بار ستون به ستون سطرها را بررسی می کنیم. اولین درایه ی غیر صفری که در یک ستون مشخص با آن مواجه می شویم را به عنوان leading entry فرضی در نظر میگیریم و اگر سطر این درایه که با متغیر "r" مشخص شده، پایین تر از "val" بود، سطر "r" را با سطر "val" جابجا می کنیم. حال بهتر است به این نکته اشاره شود که متغیر "val" در اینجا به بالاترین سطری اشاره دارد که می توانیم در آن به دنبال pivot position بگردیم و "r" نیز به سطر جاری در ستون جاری (ستون cاُم) در حلقه اشاره دارد.

پس از آنکه ماتریس ما به فرم echelon در آمد، باید آنرا به فرم REF در بیاوریم؛ این کار با تابع

make_reduced_echelonform() انجام می شود. در این تابع از سمت راست ترین pivot position سطرهای بالای آن pivot position صفر می شود و در صورتی که مقدار درایه متناظر با pivot position جاری یک نبود، آنرا با تقسیم کردن بر مقدار خودش صفر می کند.

درنهایت نیز باید ماتریس به صورت پارامتریک نمایش دهیم که این کار با تابع describe_parametric() انجام می شود. در این تابع سطر به سطر ماتریس را بررسی می کنیم و در هر سطر اگر اندیس افقی و عمودی درایه ی موجود در آن مطابق با یکی از اندیس های pivot column و pivot position بود(تمامی pivot position ها در لیست "pps" که یک لیست دو بعدی است توسط عملیات های موجود در تابع make_echelonform() ذخیره می شوند)، آن درایه را ضریبی برای متغیر پایه با اندیس ستون جاری آن (basic variable) درنظر می گیریم در غیر اینصورت مقدار آن درایه را ضریبی برای متغیر آزاد با اندیسی برابر ستون جاری در نظر می گیریم.

پس از بدست آوردن متغیرهای آزاد و پایه در یک سطر، اگر آن متغیر پایه بود سمت چپ تساوی باید نوشته شود و اگر آزاد بود سمت راست؛ بنابراین متغیرهای پایه ی یک سطر را در متغیر "lhs" ذخیره

می کنیم و متغیر های آزاد را با "rhs" کانکت می کنیم. تابع describe_parametric()، با cat کردن lhs و rhs یک معادله را برای ما فراهم می کند و در حلقه دوباره به دنبال کشف معادله ی سطر بعدی می گردد و در نهایت دو پارامتر را return میکند، یکی پاسخ دستگاه معادلات به شکل پارامتریک(متغیر "sys_eq") و دیگری "free_var" که نشاندهنده ی متغیرهای آزاد ماتریس هست.

گزارش های تصویری برای چند نمونه تست کیس در زیر آمده است:

```
Run: main
C:\Users\AVAJANG\AppData\Local\Programs\Python\Python38\python.exe "D:/Uni/term3/linear algebra/Project/LA1/main.py"
Enter number of rows and columns Respectively:
3 4
Enter row1:
0 0 1 -2
Enter row2:
1 -7 0 6
Enter row3:
-1 7 -4 2
Enter constant values:
-3 5 7
Given matrix:
 [[ 0.  0.  1. -2. -3.]
 [ 1. -7.  0.  6.  5.]
 [-1.  7. -4.  2.  7.]]
Matrix in Reduced echelon form:
 [[ 1. -7.  0.  6.  5.]
 [ 0.  0.  1. -2. -3.]
 [ 0.  0.  0.  0.  0.]]
System of linear equations:
1.0X1 = 7.0X2-6.0X4 + 5.0
1.0X3 = 2.0X4-3.0

Free variables are: X2,X4

Process finished with exit code 0
```

```
Run: main
C:\Users\AVAJANG\AppData\Local\Programs\Python\Python38\python.exe "D:/Uni/term3/linear algebra/Project/LA1/main.py"
Enter number of rows and columns Respectively:
3 3
Enter row1:
0 1 -4
Enter row2:
2 -3 2
Enter row3:
5 -8 7
Enter constant values:
8 1 1
Given matrix:
 [[ 0.  1. -4.  8.]
 [ 2. -3.  2.  1.]
 [ 5. -8.  7.  1.]]
Matrix in Reduced echelon form:
 [[ 1.   0.  -5.  12.5]
 [ 0.   1.  -4.   8. ]
 [ 0.   0.   0.   2.5]]
Equation system has no answer!

Process finished with exit code 0
```

```
C:\Users\AVAJANG\AppData\Local\Programs\Python\Python38\python.exe "D:/Uni/term3/linear algebra/Project/LA1/main.py"
Enter number of rows and columns Respectively:
3 5
Enter row1:
0 3 -6 6 4
Enter row2:
3 -7 8 -5 8
Enter row3:
3 -9 12 -9 6
Enter constant values:
-5 9 15
Given matrix:
 [[ 0.  3. -6.  6.  4. -5.]
 [ 3. -7.  8. -5.  8.  9.]
 [ 3. -9. 12. -9.  6. 15.]]
Matrix in Reduced echelon form:
 [[  1.     0.    -2.     3.     0.   -23.91]
 [  0.     1.    -2.     2.     0.    -6.98]
 [  0.     0.     0.     0.     1.     3.99]]
System of linear equations:
1.0X1 = 2.0X3-3.0X4-23.91
1.0X2 = 2.0X3-2.0X4-6.98
1.0X5 =  + 3.99

Free variables are: X3,X4

Process finished with exit code 0
```

```
C:\Users\AVAJANG\AppData\Local\Programs\Python\Python38\python.exe "D:/Uni/term3/linear algebra/Project/LA1/main.py"
Enter number of rows and columns Respectively:
4 4
Enter row1:
0 -3 -6 4
Enter row2:
-1 -2 -1 3
Enter row3:
-2 -3 0 3
Enter row4:
1 4 5 -9
Enter constant values:
9 1 -1 -7
Given matrix:
 [[ 0. -3. -6.  4.  9.]
 [-1. -2. -1.  3.  1.]
 [-2. -3.  0.  3. -1.]
 [ 1.  4.  5. -9. -7.]]
Matrix in Reduced echelon form:
 [[ 1. -0. -3. -0.  5.]
 [-0.  1.  2. -0. -3.]
 [-0. -0. -0.  1. -0.]
 [ 0.  0.  0.  0.  0.]]
System of linear equations:
1.0X1 = 3.0X3 + 5.0
1.0X2 = -2.0X3-3.0
1.0X4 = -0.0

Free variables are: X3

Process finished with exit code 0
```