

به نام خدا

فراخوانی اکسل توسط GAMS و MATLAB

سینا پورخطایی منصف^۱، مرتضی راستی برزکی^۲

۳/اردیبهشت/۱۳۹۳

(نسخه اول)

مقدمه

MATLAB یک محیط نرم افزاری برای انجام محاسبات عددی و یک زبان برنامه‌نویسی نسل چهارم است. هسته MATLAB برای سرعت و کارایی بالا به زبان برنامه‌نویسی C نوشته شده‌است ولی رابط گرافیکی آن به زبان JAVA پیاده سازی گشته‌است. GAMS یک محیط برنامه نویسی مدل‌های برنامه‌ریزی ریاضی با قابلیت بالاست. از GAMS برای حل مسائل برنامه‌ریزی خطی (LP)، برنامه‌ریزی غیر خطی (NLP)، برنامه‌ریزی صحیح مختلط (MIP)، برنامه‌ریزی خطی صحیح مختلط (MINLP) و ... استفاده می‌کنند.

اغلب در تحقیقات و مسائل کاربردی اتفاق می‌افتد نرم افزاری مانند اکسل توسط GAMS یا MATLAB فراخوانی و اجرا شود.

بخش اول: فراخوانی اکسل توسط GAMS

در این بخش به توضیح مختصری از چگونگی استفاده از فایل GDX (توضیحات بیشتر در یادداشت مربوط به ارتباط GAMS و MATLAB) در GAMS به منظور خواندن اطلاعات از EXCEL و نوشتن خروجی در آن خواهیم پرداخت. GAMS توسط فایل GDX با اکسل ارتباط برقرار می‌کند. فایل GDX دارای ابزارهای مختلفی برای استفاده است. از این رو از ابزار GDXXRW به منظور خواندن و نوشتن اطلاعات در اکسل استفاده می‌شود.

^۱ کارشناسی ارشد، دانشکده مهندسی صنایع و سیستم‌ها، دانشگاه صنعتی اصفهان، s.pourkhataei@in.iut.ac.ir

^۲ استادیار، دانشکده مهندسی صنایع و سیستم‌ها، دانشگاه صنعتی اصفهان، <http://rasti.iut.ac.ir>

خواندن اطلاعات از اکسل

گام اول - ایجاد فایل GDx

فرم کلی این دستور به شرح زیر است:

```
gdxxrw Excel_FileName/Excel_filePath.xlsx Input_Information
```

در قسمت Excel_FileName نام مسیر فایل اکسلی که می‌خواهیم اطلاعات را از آن وارد کنیم نوشته می‌شود و اگر فایل اکسل در پوشه دایرکتوری نرم افزار GAMS باشد تنها نام فایل اکسل کافی است. دقت شود که پوشه دایرکتوری GAMS متفاوت از مسیر نصب نرم افزار GAMS است. این پوشه معمولاً در مسیر زیر قرار دارد:

C:\Users\UserName\Documents\GAMSdir\projdir

که در این مسیر، قسمت **UserName** برای هر کاربر متفاوت است و کاربر باید نامی را که برای User خود انتخاب کرده قرار دهد. اگر از ورژن ۲۰۱۰ به بعد مجموعه Office استفاده می‌کنید باید از پسوند 'xlsx' استفاده شود در غیر این صورت باید از پسوند 'xls' استفاده شود. قسمت Input_Information شامل نام مجموعه یا پارمتر، محدوده‌ی مورد نظر و ابعاد داده است. ابزار GDXXRW باید توسط نرم افزار GAMS فراخوانی شود که این کار از طریق دستور \$call قبل از این دستور صورت می‌گیرد. به منظور روشن‌تر شدن بحث فرض کنید پارامتری داریم به نام d که دارای دو بُعد است و مقدار آن در شکل ۳ نمایش داده شده است. عناصر A2:A3 و عناصر B1:D1 بیانگر اندیس‌های این پارامتر می‌باشد که فرض می‌کنیم از قبل آن‌ها را تعریف کرده‌ایم. در هنگام فراخوانی، این اندیس‌ها، جزء ماتریس مقادیر قرار نمی‌گیرند و فایل GDx به صورت خودکار قابلیت تشخیص اندیس‌ها را از مقادیر پارامترها دارد، حتی در مواردی که اندیس دارای مقدار عددی باشد.

	A	B	C	D	E
1		a1	a2	a3	
2	i1	1	2	3	
3	i2	4	5	6	
4					

شکل ۱

از دستور زیر به منظور فراخوانی از اکسل استفاده می‌کنیم:

```
$call gdxxrw test1.xlsx par=d rng=A1:D3 rdim=1 cdim=1
```

همانطور که مشاهده می‌کنید در قسمت Input_Information ابتدا نوع داده را مشخص کردیم. هنگامی که داده پارامتر باشد از عبارت 'par' و هنگامی که مجموعه باشد از عبارت 'dset' استفاده می‌کنیم. محدوده‌ی مقادیر این پارامتر با عبارت 'rng' مشخص می‌شود. عبارت 'rdim' ابعاد پارامتر در سطر و عبارت 'cdim' ابعاد پارامتر در ستون را نشان می‌دهد. همانطور که می‌بینید هم در سطر و هم در ستون ابعاد پارامتر یک است.

نکته: اگر فرض کنیم که فرم پارامتر d در فایل اکسل همانند شکل ۴ باشد تفاوت آن با حالت قبل در ابعاد پارامتر و محدوده‌ی مقادیر می‌باشد. در این حالت ستون دارای هیچ بُعدی نیست ولی سطرها دارای دو بُعد هستند.

	A	B	C	D
1	i1	a1	1	
2	i1	a2	2	
3	i1	a3	3	
4	i2	a1	4	
5	i2	a2	5	
6	i2	a3	6	
7				

شکل ۲

دستور آن به صورت زیر است:

```
$ call gdxrw test1.xlsx par=d rng=A1:C6 rdim=2
```

هر دو حالت معادل همدیگر هستند.

نکته:

یکی از ویژگی‌های جالب ابزار gdxrw این است که اگر در یک sheet از اکسل تنها یک پارامتر و یا یک مجموعه را داشته باشیم، به منظور تعیین محدوده مورد نظر کافی است تنها نقطه شروع را وارد کنیم. به عبارتی در شکل ۳ و ۴ عبارت rng=A1 نتایج یکسانی با حالت‌های قبلی دارد. این ویژگی در مواردی بیشتر مورد استفاده قرار می‌گیرد که کاربر بخواهد ابعاد مسئله را دائماً تغییر دهد. در این حالت نیاز نیست به صورت دستی در هر بار از تغییر ابعاد مجموعه یا پارامتر به صورت دستی محدوده‌ی آن را نیز اصطلاح کرد و با این کار به صورت خودکار محدوده‌ی مورد نظر به‌نگام می‌شود. در این حالت لازم است که کاربر مجموعه‌ها و یا پارامترهای خود را در sheet های مستقل قرار دهد.

-استفاده از فایل text به منظور خوانایی بیشتر

فرض کنید که قصد داریم یک مجموعه و یک پارامتر را از فایل اکسل فراخوانی کنیم. اگر بخواهیم به روش گفته شده در قبل این کار را انجام دهیم دستور زیر را باید وارد کنیم:

```
gdxxrw test2.xlsx dset=i rng=sheet1!A1 rdim=1 par=data rng=sheet2!A1 rdim=1
```

همانطور که می بینید هنگامی که هنگامی که تعداد مجموعه ها و یا پارامترها افزایش یابد خط دستور بسیار طولانی می شود. در این صورت بهتر است از طریق یک فایل text آن را به چندین خط تبدیل کنیم که هم باعث افزایش خوانایی و سهولت آن می شود. به همین منظور فرم کلی این حالت به شکل زیر است:

```
$onecho > text_FileName.txt
```

```
Input_Information
```

```
$offecho
```

```
$call gdxxrw excel_FileName/excel_filePath.xlsx @text_FileName.txt
```

دستور \$onecho و \$offecho همواره ثابت است. در قسمت text_fileName یک نام دلخواه برای تولید فایل text انتخاب می کنیم. در قسمت Input_Information اطلاعات ورودی هر داده را وارد می کنیم منتهی هر داده را در یک سطر می نویسیم.

در مثال قبلی فرم جدید فراخوانی از اکسل به صورت زیر است:

```
$onecho > text1.txt
```

```
dset=i rng=sheet1!A1 rdim=1
```

```
par=data rng=sheet2!A1 rdim=1
```

```
$offecho
```

```
$call gdxxrw test2.xlsx @text1.txt
```

گام دوم: فراخوانی فایل GDX توسط GAMS

هنگامی که گام اول به اتمام رسید، در خط بعد از آن، از دستور \$GDXIN به منظور فراخوانی فایل GDX استفاده می کنیم. فرم این دستور به شکل زیر است:

```
$call GDXIN excel_FileName/excel_FilePath.gdx
```

دقت شود که نام فایل GDX دقیقاً باید معادل نام فایل اکسلی باشد که داده ها را از آن فراخوانی می کنیم. اگر فایل اکسل در داخل مسیر C:\Users\UserName\Documents\GAMSdir\projdir باشد، تنها استفاده از نام فایل کافی است. ولی اگر فایل

اکسل در مسیر متفاوتی باشد این نکته را باید در نظر داشت که فایل GDX تولید شده توسط اکسل حتماً در داخل مسیر C:\Users\UserName\Documents\GAMSdir\projdir ریخته می‌شود. بنابراین باید برای فراخوانی فایل GDX دستور زیر را وارد کنیم:

```
$call GDXIN C:\Users\UserName\Documents\GAMSdir\projdir \excel_FileName.gdx
```

گام سوم- بارگذاری مجموعه‌ها یا پارامترها

در این گام با استفاده از دستور \$LOAD بعد از تعریف مجموعه و یا پارامتر، آن‌ها را بارگذاری می‌کنیم. فرم این دستور به شکل زیر است:

```
$LOAD SetName/ParameterName
```

به طور معمول پارامترها بعد از مجموعه‌ها تعریف می‌شود. از همین رو در انتهای بارگذاری پارامترها دستور زیر را به منظور بستن فایل GDX وارد کنید:

```
$GDXIN
```

نوشتن اطلاعات در اکسل

گام اول- ایجاد فایل GDX

در این گام با دستور زیر به ایجاد فایل GDX می‌پردازیم:

```
Execute_unload " GDX_FileName/GDX_FilePath.gdx "
```

این دستور در خط آخر مدل نوشته می‌شود و به منظور ایجاد یک فایل GDX از خروجی مدل استفاده می‌شود. تعیین نام فایل GDX به صورت دلخواه است. اگر به جای مسیر فایل، نام فایل را قرار دهیم GDX تولید شده در مسیر زیر ریخته می‌شود.

```
C:\Users\UserName\Documents\GAMSdir\projdir
```

گام دوم- فراخوانی فایل GDX و نوشتن اطلاعات در اکسل

در این گام ابتدا فایل GDX فراخوانی می‌شود و سپس درون یک فایل اکسل با نام دلخواه نوشته می‌شود. فرم کلی این دستور به شکل زیر است:

```
Execute ' gdxrw.exe GDX_FilePath.gdx o=excel_FilePath.xlsx OutPut_information '
```

در قسمت `GDX_FilePath` نام مسیر فایل `GDX` تولید شده در گام اول را وارد می‌کنیم. اگر فایل `GDX` درون پوشه دایرکتوری `GAMS(GAMSdir)` باشد تنها نوشتن نام فایل کافی است. در قسمت `excel_FilePath` نام مسیر فایل اکسل که می‌خواهیم اطلاعات در آن نوشته شود قرار می‌گیرد. اگر به جای مسیر تنها نام فایل را ذکر کنیم، فایل اکسل تولید شده در پوشه دایرکتوری `GAMS` قرار می‌گیرد. در قسمت `OutPut_Information` اطلاعات لازم از عناصری که می‌خواهیم در اکسل وارد شود را وارد می‌کنیم. در این قسمت تنها عنصر مورد نظر و محدوده‌ی آن را مشخص می‌کنیم. برای متغیرها از عبارت `'var'` به منظور تعیین نوع آن استفاده می‌کنیم. لازم به ذکر است که تمامی این دستور باید در یک خط نوشته شود.

ارائه مثال

مدل حمل و نقل ارائه شده در یادداشت ارتباط `GAMS` و `MATLAB` را در نظر بگیرید. قصد داریم که مجموعه‌ها و پارامترها را از داخل اکسل فراخوانی کنیم و نتایج خروجی بر روی فایل اکسل دیگری بریزیم. فرض کنید که فایلی که می‌خواهیم از روی آن اطلاعات را وارد کنیم در مسیر `"D:\data\excel_data.xlsx"` باشد. کد دستوری در `GAMS` به شرح زیر است:

```
$onecho >data_text.txt
dset=i rng=i!A1 rdim=1
dset=j rng=j!A1 cdim=1
par=S rng=Supply!A1 rdim=1
par=D rng=Demand!A1 cdim=1
par=C rng=Cost!A1 rdim=1 cdim=1
$offecho
$CALL GDXXRW.EXE D:\data\excel_data.xlsx @data_text.txt
$GDXIN C:\Users\UserName\Documents\GAMSdir\projdir\excel_data.gdx
```

Sets

```
i   canning plants
j   markets ;
$LOAD i,j
```

Parameters

```
S(i)  capacity of plant i in cases
D(j)  demand at market j in cases
C(i,j) distance in thousands of miles ;
$LOAD S,D,C
$GDXIN
```

```
Scalar f  freight in dollars per case per thousand miles /90/ ;
Parameter CC(i,j)  transport cost in thousands of dollars per case ;
CC(i,j) = f * C(i,j) / 1000 ;
```

Variables

```
x(i,j)  shipment quantities in cases
z        total transportation costs in thousands of dollars ;
Positive Variable x ;
```

```

Equations
cost          define objective function
supply(i)     observe supply limit at plant i
demand(j)     satisfy demand at market j ;

cost ..      z  =e=  sum((i,j), CC(i,j)*x(i,j)) ;
supply(i) ..  sum(j, x(i,j))  =l=  S(i) ;
demand(j) ..  sum(i, x(i,j))  =g=  D(j) ;

Model transport /all/ ;
Solve transport using lp minimizing z ;

execute_unload "D:\data\outGDX.gdx"
execute 'gdxrw.exe D:\data\outGDX.gdx o= D:\data\OutPut.xlsx var=z.1
rng=z!A1 var=x.1 rng=x!A1'

```

نکته:

لازم است دستور execute در نرم افزار GAMS تنها در یک خط نوشته شود. ولی به علت بزرگتر بودن دستور از عرض صفحه نتوانستیم آن را در یک خط بنویسیم.

بخش دوم: فراخوانی اکسل توسط MATLAB

گاهی اوقات لازم است که اطلاعات مورد نیاز از درون فایل اکسل به متلب فراخوانی و یا نتایج خروجی از متلب به فایل اکسل ریخته شود. در این حالت می‌توان از تابع xlsread() برای خواندن اطلاعات و از تابع xlswrite() به منظور ریختن اطلاعات در اکسل توسط نرم افزار متلب استفاده کرد.

۱- خواندن اطلاعات از اکسل:

فرم کلی تابع xlsread() به صورت زیر است:

```
xlsread('filename','sheet','range',...)
```

همانطور که مشاهده می‌کنید این تابع دارای آرگومان‌های متعددی می‌باشد ولی اصلی‌ترین آرگومان‌های آن به شرح زیر است:

:FileName

یک رشته است که نام مسیر فایل اکسل و یا در صورتی که فایل اکسل در پوشه جاری متلب باشد نام فایل اکسلی است که می‌خواهیم اطلاعات را از آن وارد کنیم.

:sheet

یک رشته است که نام صفحه‌ای (sheet) را که می‌خواهیم اطلاعات را از آن وارد کنیم نوشته می‌شود.

:range

محدوده‌ی داده‌های مورد نظر را در این قسمت وارد می‌کنیم.

به طور مثال داریم :

```
xlsread('D:\data\excel1.xlsx','sheet1','A1:C6')
```

۲- نوشتن اطلاعات در اکسل:

فرم کلی تابع `xlswrite()` به صورت زیر است:

```
xlswrite('filename', A, 'sheet', 'range',...)
```

آرگومان‌های این تابع بسیار شبیه تابع `xlsread()` است و از ذکر مجدد آن خودداری می‌کنیم. تنها تفاوت آن در آرگومان دوم است. آرگومان دوم (A) بیان‌گر آن عنصری است که می‌خواهیم درون فایل اکسل نوشته شود.

به طور مثال داریم:

```
xlsread('D:\data\OutPut.xlsx',data,'sheet1','A1:C6')
```

نکته= در آرگومان اول تابع `xlswrite()`، اگر فایل اکسل موجود باشد اطلاعات را بر روی آن فایل می‌ریزد ولی اگر فایل اکسلی موجود نباشد، متلب به صورت خودکار یک فایل اکسل با نام ذکر شده می‌سازد و اطلاعات را درون آن می‌ریزد.

پایان