두번째: SAM

강성원

KEI

2016.08.04

개관

SAM 구축 순서

1. 산업 mapping

2. IO aggregation

3. SAM

4.Set

차 례

순서

1. 산업 mapping 확정
2. IO aggregation
3. SAM 구축
4. set 점검

차 례

1. 산업 mapping 확정

384개 기본부문 → 37개 산업 (표준모형) → 7개 산업 (Prototype)
(indcode_20161202.xlsx, indcode_20161202.csv)

basecode_row	sector_row	basecode_column	sector_row	sector_ind	sector_name	Group_ind	Group_name	Va_index	Va_name	Va_dict	FD_index	FD_name	FD_dict	BR_ind	BR_name
1	35	1	35	1ELEC		1	1Transform	38	idinput1	소계	38	Dint	중간수요계		1ELEC
2	35	2	35	2GAS		1	1Transform	39	PTAXin	순생산물세	39	HE	민간소비지출		2GASHeat
3	35	3	35	3HEAT		1	1Transform	40	Resin	잔폐물발생	40	GE	정부소비지출		2GASHeat
4	35	4	35	4ROIL		2	Ind	41	idinput2	중간투입계	41	HI	민간고정 자본형성		3OIL
5	35	5	35	5CoalPro		2	Ind	42	Payroll	피용자보수	42	GI	정부고정 자본형성		4COAL
6	35	6	35	6Coal		2	Ind	43	Surplus	영업잉여	43	ST	재고증감		4COAL
7	35	7	35	7Oil		2	Ind	44	Deprec	고정자본소모	44	Gold	귀중품 순취득		3OIL
8	35	8	35	8LNG		2	Ind	45	PTAXetc	기타생산세	45	EX	수출		2GASHeat
9	35	9	35	9Mining		2	Ind	46	VA	부가가치계	46	Dfin	최종수요계		5ENIT
10	35	10	35	10IS		2	Ind	47	Tinput	총투입계	47	Dtotal	총수요계		5ENIT
11	35	11	35	11Cement		2	Ind				48	Qtotal	총산출		5ENIT
12	35	12	35	12Orgchem		2	Ind				49	Qself	자가공정산출		5ENIT
13	35	13	35	13WoodPaper		2	Ind				50	Imp	수입		6NEINT
14	35	14	35	14FiberLeather		2	Ind				51	Resout	잔폐물발생액		6NEINT
15	35	15	35	15Mineral		2	Ind				52	Stotal_b	총공급계 (기초가격)		5ENIT
16	35	16	35	16nonISmetal		2	Ind				53	PTAXd	생산물세(국산)		6NEINT
17	35	17	35	17Machine		2	Ind				54	PTAXim	생산물세(수입)		6NEINT
18	35	18	35	18Electro_e		2	Ind				55	Psub	생산물보조금 (차감)		6NEINT
19	37	19	37	19Electro_ne		2	Ind				56	Cmargin	도소매마진		6NEINT
20	37	20	37	20Electro_sig		2	Ind				57	Tmargin	화물운임		6NEINT
21	37	21	37	21semicon		2	Ind				58	Contotal	가격전환계		6NEINT
22	37	22	37	22Auto		2	Ind				59	Stotal_p	총공급계		6NEINT
23	37	23	37	23Ship		2	Ind								6NEINT
24	37	24	37	24Food		2	Ind								6NEINT
25	33	25	33	25MissManu		2	Ind								6NEINT
26	6	26	6	26Const		2	Ind								6NEINT
27	6	27	6	27Rail		3	Transport								5ENIT
28	7	28	7	28Road		3	Transport								5ENIT
29	8	29	8	29Air		3	Transport								5ENIT
30	9	30	9	30Marine		3	Transport								5ENIT
31	9	31	9	31MissTrans		3	Transport								5ENIT
32	9	32	9	32Housing		4	Housing								6NEINT
33	9	33	9	33Commercial		5	Commercial								6NEINT
34	9	34	9	34Public		6	Public								6NEINT
35	35	35	35	35Agri		7	Agri								7Agri
36	35	36	35	36Waste		8	Waste								6NEINT
37	24	37	24	37FF		2	Ind								5ENIT

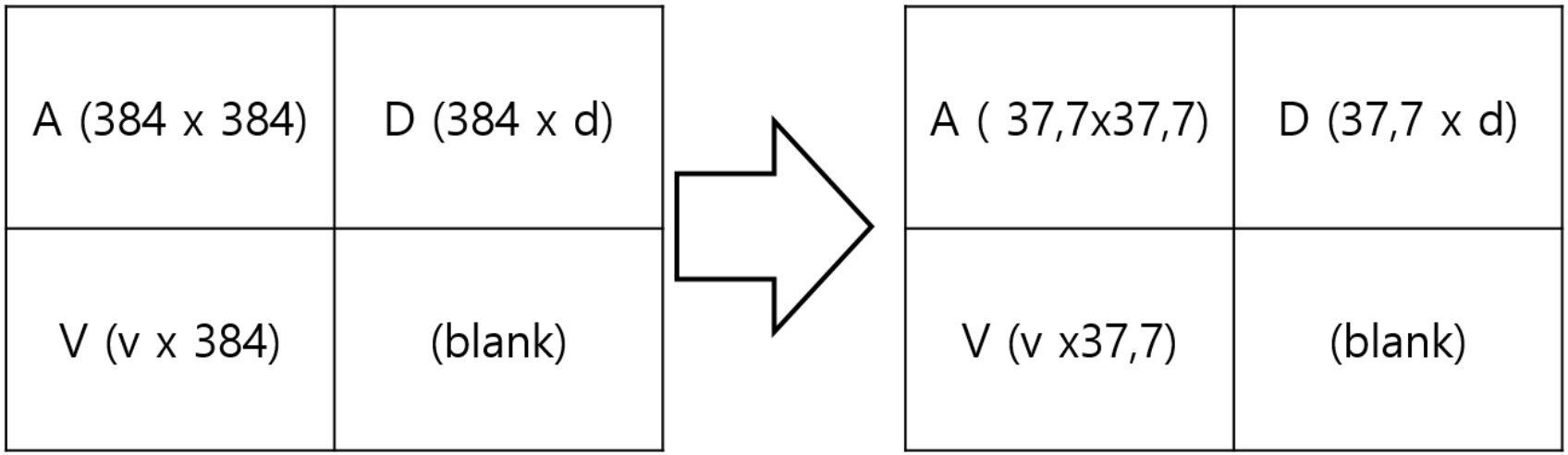
차 례

2. IO aggreation

IO aggregation : 차원을 줄인 IO 만들기 (agg_1202.r, agg_ghg_1202.r)

- agg_1202.r : IO 차원 줄이기
  - input: IOT_b.csv (기초가격산업연관표), indcode_20161202.csv (산업 mapping)
  - output: IO_model_1202.csv (37개 산업 IO), IO_B_1202.csv (7개 산업 IO)
- agg_ghg_1202.r: 온실가스 IO 차원 줄이기
  - input: GHGIO.csv (온실가스IO, GTAP-K), indcode_20161202.csv (산업 mapping)
  - output: GHG_model_p_1202.csv (37개 산업 온실가스 IO), GHG_BR_p_1202.csv (7개 산업 온실가스 IO)

IO aggregation



$$\bar{a}_{I,J} = \sum_{i \in I} \sum_{j \in J} a_{i,j} \qquad \bar{v}_{v,J} = \sum_{j \in J} v_{v,j} \qquad \bar{d}_{I,d} = \sum_{i \in I} d_{i,d}$$

IO aggregation:agg_1202.r

```
## Step 1. Data preperation
#(i) loading
IOT_b=read.csv(file="IOT_b.csv",header=T, as.is=T)
sector_ind=read.csv(file="indcode_20161202.csv",header=T, as.is=T)

## Step 2. Rowsum:  merge and obtain rowsum using aggregate function
IOT_b_sec=merge(IOT_b,row_ind, by="basecode_row", all=T)
IOT_b_37=aggregate(IOT_b_sec[,4:(dim_IOT_b[2])],
list(IOT_b_sec$sector_row),FUN=sum)

## Step 3. Column sum
#(i) Traspose rowsum
T_IOT_b_37=data.frame(t(IOT_b_37))
T_IOT_37_col=aggregate(T_IOT_b_37[,1:(nsector+nva)],
list(T_IOT_b_37$basecode_col),FUN=sum)

## Step 4. obtain IO table
#(i)obtain transpose of column sum
IOT_37=data.frame(t(T_IOT_37_col))
#(ii) add column names
colnames(IOT_37)[1:nsector]=sec_group[(1:37),2]
colnames(IOT_37)[(nsector+1):(nsector+nfd)]=fd_ind[,2]

write.csv(IOT_37, file="IO_model_1202.csv")
```

차 례

3. SAM :IO의 item을 이용해서 SAM 만들기

1. 가계-정부 간 소득이전: 소득세, 이전지출
2. 부문 별 저축 확정
   - 가계저축 = 요소소득 + 이전소득 - 민간소비
   - 정부저축 = 조세수입 - 정부소비 - 이전지출
   - 해외저축 = 수입액 - 수출액

A (7 x 7)	D
V	(blank)

Table 1—The Basic SAM structure used in the CGE model

	Expenditures								
Receipts	Activities	Commodities	Factors	Households	Enterprises	Government	Savings-Investment	Rest of the World (ROW)	Total
Activities		Marketed outputs		Home-consumed outputs					Activity income (gross output)
Commodities	Intermediate inputs	Transaction costs		Private consumption		Government consumption	Investment	Exports	Demand
Factors	Value-added							Factor income from ROW	Factor income
Households			Factor income to households	Interhousehold transfers	Surplus to households	Transfers to households		Transfers to households from ROW	Household income
Enterprises			Factor income to enterprises			Transfers to enterprises		Transfers to enterprises from ROW	Enterprise income
Government	Producer taxes, value-added tax	Sales taxes, tariffs, export taxes	Factor income to government, factor taxes	Transfers to government, direct	Surplus to Government, direct enterprise taxes			Transfer to Government from ROW	Government income
Savings-Investment				Household savings	Enterprise savings	Government savings		Foreign savings	Savings
Rest of the World (ROW)		Imports	Factor income to ROW		Surplus to ROW	Government transfers to ROW			Foreign exchange outflow
Total	Activity	Supply expenditures	Factor expenditures	Household expenditures	Enterprise expenditures	Government expenditures	Investment	Foreign exchange inflow	

SAM

	산업	재화	생산요소	가계	정부	조세	저축-투자	해외
산업		S.(산업)						
재화	A			민간소비(D)	정부소비(D)		고정자본형성(D) +재고(D) +귀금속(D)	수출(D)
생산요소	영업잉여(V) +감가상각(V) 피용자보수(V)							
가계			S.(영업잉여) +S.(감가상각) S. 피용자보수		이전지출*			
정부						S. 생산세(V) -S.생산보조금(V) S. 관세(D) + S. 수입세(D) 소득세*		
조세	생산세(V) -생산보조금(V)	관세(D)+ 수입세(D)		소득세 *				
저축-투자				가계저축**	정부저축**			해외저축**
해외		수입(D)						

S. (변수) 는 합계를 의미

*: 소득세, 이전지출은
국민계정, '제도부문별 소득계정'

**가계저축 = 요소소득+이전소득-민간소비-소득세
**정부저축 = 조세수입-정부소비-이전지출
** 해외저축 =S.(수입)-S. (수출)

input-output

input

1. Aggregated IO
   - IO_model_1202.csv (37개 산업 IO)
   - IO_B_1202.csv (7개 산업 IO)
2. Aggregated GHG IO
   - GHG_model_p_1202.csv (37개 산업 온실가스IO)
   - GHG_BR_p_1202.csv (7개 산업 온실가스IO)

output

1. 온실가스가 반영된 SAM
   - b_sam_36_g_pos_1202.csv (37개산업)
   - b_sam_br_g_1202.csv(7개산업)
2. 온실가스가 반영되지 않은 SAM
   - b_sam_36_ng_pos_1202.csv(37개산업)
   - b_sam_br_ng_1202.csv(7개산업)

Process

1. samconst_2010_pos_1202.r

SAM 구축:samconst_2010_pos_1202.r

```
## load sam aggregation function
source("agg_1202.r")
source("agg_ghg_1202.r")
source("sam_2010.r")

#STEP 1: Size.SAM Determine the size of SAM
ind=37;green=1;fac=2;h=1;gov=1;Nres=1;tax=4;S_I=1;ROW=1
Size.Sam=c(ind,green,fac,h,gov,Nres,tax,S_I,ROW)

#STEP 2: data.out : NON I-O data
YTAX=83753*1000
TRANSFER=39046*1000
data.out=c(YTAX,TRANSFER)

# STEP 3: I-O data ##load I-O data (model)
IO_model=read.csv("IO_model_1202.csv",header=T, as.is=T)
DIO=as.matrix(IO_model2)

#STEP 4: SAM construction
SAM_model=data.frame(SAM_agg_basic(Size.Sam,data.out,DIO))
model_SAM_name=c(m_Activity_name,"CO2-a",m_commodity_name,"CO2-c",
factor_name,Inst_name1,tax_name,Inst_name2)
rownames(SAM_model)=model_SAM_name
colnames(SAM_model)=model_SAM_name
```

b_sam_br_g_1202.csv

	ELEC-a	GASHeat-a	OIL-a	COAL-a	ENIT-a	NEINT-a	AGRI-a	CO2-a	ELEC-c	GASHeat-c	OIL-c	COAL-c	ENIT-c	NEINT-c	AGRI-c	CO2-c	Labor	Capital	Household	GoV	NRES	Ptaxin	Ptaxetc	Tarrif	YTAX	S-I	ROW
ELEC-a	0	0	0	0	0	0	0	0	41534.49	0	0	0	0	0	0	0	0	0	0	2427.788	0	0	0	0	0	0	0
GASHeat-a	0	0	0	0	0	0	0	0	0	30161.79	0	0	0	0	0	0	0	0	0	116.0256	0	0	0	0	0	0	0
OIL-a	0	0	0	0	0	0	0	0	0	0	103050.7	0	0	0	0	0	0	0	0	49.77111	0	0	0	0	0	0	0
COAL-a	0	0	0	0	0	0	0	0	0	0	0	6912.343	0	0	0	0	0	0	0	1.195342	0	0	0	0	0	0	0
ENIT-a	0	0	0	0	0	0	0	0	0	0	0	0	532006.5	0	0	0	0	0	0	1692.023	0	0	0	0	0	0	0
NEINT-a	0	0	0	0	0	0	0	0	0	0	0	0	0	2387212	0	0	0	0	0	749.6459	0	0	0	0	0	0	0
AGRI-a	0	0	0	0	0	0	0	0	0	0	0	0	0	0	52969.02	0	0	0	0	55.90367	0	0	0	0	0	0	0
CO2-a	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5092.352	0	0	0	0	0	0	0	0	0	0	0
ELEC-c	556.757	64.474	815.588	72.901	7486.617	24304.22	272.211	0	0	0	0	0	0	0	0	0	0	0	7975.42	0	0	0	0	0	0	0	83.761
GASHeat-c	8575.153	20534.6	470.029	0.785	3160.66	8720.109	9.565	0	0	0	0	0	0	0	0	0	0	0	7870.849	0	0	0	0	0	0	455.704	18.034
OIL-c	2300.089	681.308	83242.31	67.988	52686.65	22582.39	1632.669	0	0	0	0	0	0	0	0	0	0	0	10533.37	0	0	0	0	0	0	2220.658	40993.88
COAL-c	8892.316	469.276	0	5046.834	8129.638	286.481	103.351	0	0	0	0	0	0	0	0	0	0	0	81.243	0	0	0	0	0	0	-909.442	23.69
ENIT-c	639.452	61.631	2806.869	355.885	239669.9	247991.8	1451.984	0	0	0	0	0	0	0	0	0	0	0	23337.58	0	0	0	0	0	0	2139.022	140680.3
NEINT-c	7651.798	1518.722	7589.852	808.42	93588.16	1034045	17100.04	0	0	0	0	0	0	0	0	0	0	0	527027.6	183108.5	0	0	0	0	0	370461.2	449400.4
AGRI-c	1.268	0.281	1.534	2.866	1430.396	41220.73	3685.653	0	0	0	0	0	0	0	0	0	0	0	14509.49	0	0	0	0	0	0	1949.315	776.428
CO2-c	2427.788	116.0256	49.77111	1.195342	1692.023	749.6459	55.90367	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Labor	3438.079	925.373	1112.996	141.264	46078.63	480825.7	3828.25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Capital	9153.961	3779.67	5012.656	441.533	65433.41	487873.2	23425.29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Household	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	536350.3	595119.7	0	39046	1075.434	0	0	0	0	0	0
GoV	0	0	0	0	0	0	0	5092.352	0	0	0	0	0	0	0	0	0	0	0	0	0	140861.3	13654.09	0	83753	0	0
NRES	-47.827	-5.84	369.524	-1.204	6563.552	-2947.48	-2.314	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Ptaxin	263.253	2110.82	1487.932	-26.574	6848.875	30835.62	486.808	0	0	0	0	0	0	0	0	0	0	0	46452.57	0	0	0	0	0	0	31724.56	20677.39
Ptaxetc	110.188	21.479	141.412	1.645	930.019	11473.83	975.517	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Tarrif	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
YTAX	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	83753	0	0	0	0	0	0	0	0
S-I	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	450050.3	16113.83	2853.031	0	0	0	0	0	-60976.1
ROW	0	0	0	0	0	0	0	0	97.461	19653.69	113890.6	15211.04	127127.8	305088.1	10608.94	0	0	0	0	0	0	0	0	0	0	0	0

차 례

4.set

- Contingent Set의 정의가 Data에 따라 변화→ Contingent set이 활용된 program code는 Data 변화에 영향

- *: Contingent Set의 변화에 따라 영향을 받을 수 있는 부분 (Agritoy_recursive.gms)
  1. Declaration (set)*
  2. Data Loading*
  3. Equations*
  4. Calibration(Parameter,Initial value)*
  5. Model Statement
  6. Initialization
  7. Solve Statement
  8. Save and dispatch

- Agritoy_recursive.gms의 경우 contingency set의 구성이 sector 변환 이전과 동일하여서 SAM과 CGE의 일관성이 유지

Contingency Set

Contingent Set: joint set 으로 한 집합의 element에 따라 다른 집합의 구성요소가 달라지는 set

```
FD_C(C,FD)
/
GASHeat-c.(S-I )
OIL-c.(S-I )
COAL-c.(S-I )
ENIT-c.(S-I )
NEINT-c.(GoV,S-I )
Agri-c.(S-I )
/

ComMENCN(C)$(ENCN(C))..XA(C)=g=sum(A$XAPA(C,A),XAP(C,A))
+sum(H$XACH(H,C),XAC(C,H))+sum(FD$FD_C(C,FD),XAF(FD,C));
```

Set Declaration의 자동화 (setwritting_2015_br_1202.r)

```
sam_br=read.csv(file="b_sam_br_ng_1202.csv",header=T,as.is=T)
ghg=read.csv(file="GHG_BR_p_1202.csv",header=T,as.is=T)
filename="set_br_20151204.txt"
sink(file=filename)

#Non household (Domestic) Final demand: FD
FD=c("GoV","S-I")

#Final demand mix for non household institutions
XFA=sam_br[Commodity,match(FD,colnames(sam_br))]
T_XFA=data.frame(t(XFA))
colnames(T_XFA)=rownames(XFA)
Positive.fin.demand=lapply(T_XFA,function(x){rownames(T_XFA)[x!=0]})
Positive.fin.demand=Positive.fin.demand[mapply(FUN=length,Positive.fin
XFA.A={}
for (i in (1:length(Positive.fin.demand))){
  XFA.A_i=paste(paste(names(Positive.fin.demand)[i],HC.com(Positive.fi
  XFA.A=rbind(XFA.A,XFA.A_i)
}
cat("FD_C(C,FD)",sep="\n")
cat("/",sep="\n")
cat(XFA.A,sep="\n")
cat("/",sep="\n")
```

Set Declaration의 자동화: output (set_br_20151211.txt)

```
....
XEPA(C,A)
/
GASHeat-c.(ELEC-a,GASHeat-a,OIL-a,COAL-a,ENIT-a,NEINT-a,Agri-a )
OIL-c.(ELEC-a,GASHeat-a,OIL-a,COAL-a,ENIT-a,NEINT-a,Agri-a )
COAL-c.(ELEC-a,GASHeat-a,COAL-a,ENIT-a,NEINT-a,Agri-a )
/
FD(ACT) /GoV,S-I /
FD_C(C,FD)
/
GASHeat-c.(S-I )
OIL-c.(S-I )
COAL-c.(S-I )
ENIT-c.(S-I )
NEINT-c.(GoV,S-I )
Agri-c.(S-I )
/
.....
Alias(GC,GCP);
Alias(ENC,ENCPP);
ACNT(ACT)=yes;
ACNT('Total')=no;
```

...

감사합니다.