NREL
NATIONAL RENEWABLE ENERGY LABORATORY

Augmenting blueNOTE data operations for SLiDE

Caroline L. Hughes

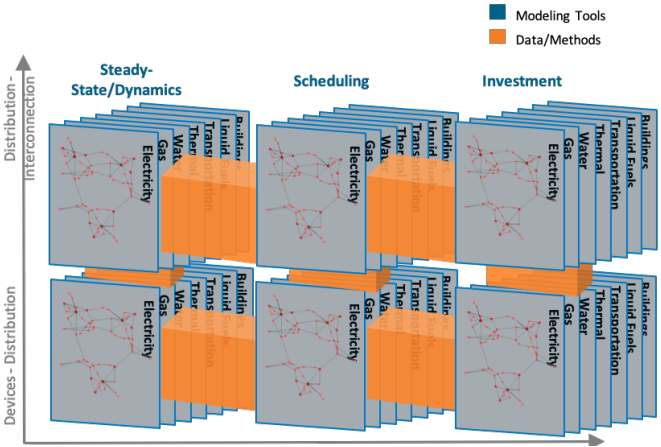
May 27, 2020

Motivation

- We seek to remove the overly-simplistic and static assumptions concerning economic boundary conditions by developing a model of the economy and coupling that model with detailed, engineering-focused energy system models.
- Expand the user-base of the blueNOTE data set to Julia-focused developers

Scalable Integrated Infrastructure Planning (SIIP)

NREL initiative housed under the Scientific Computing and Energy Analysis (SCEA) directorate to unify models under one umbrella (https://github.com/NREL-SIIP)



Scalable Linked Dynamic Equilibrium (SLiDE)

Goal: Use blueNOTE to build an open-source computable general equilibrium model for Scalable Integrated Infrastructure Planning (SIIP) in Julia

What's in a name?

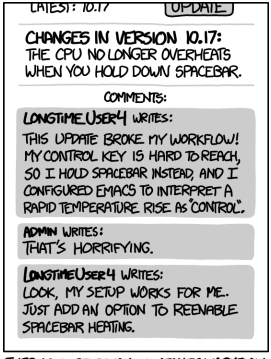
- *Scalable* in both region and sectoral resolution
- *Linked* with SIIP models (ReEDS[1], Tempo[2], ...), similar to USREP-ReEDS[3]
- *Dynamic* - either recursive or intertemporal - built into the model at this time; still working on recursive dynamics
- Computable General *Equilibrium*

[1] S. Cohen et al., NREL/TP-6A20-72023 (National Renewable Energy Laboratory, Golden, CO, 2019).
[2] M. Muratori et al., Renew Sustain Energ Rev **119** (2020).
[3] T. Carson et al., Clim Chang Econ **9** (2018).

Why Julia[6]?

Challenges:

- Learning curve
- Under rapid development
  - Capabilities lacking
  - Frequent updates without backwards compatibility
- Syntax not as clean as GAMS[4]. No MPSGE[5] ..yet?



https://xkcd.com/1172/
(via the Julia Slack channel)

[6] J. Bezanson et al., SIAM Rev **59** (2017).
[5] T. F. Rutherford, Comput Econ **46** (1999).

Why Julia[6]?

Challenges:

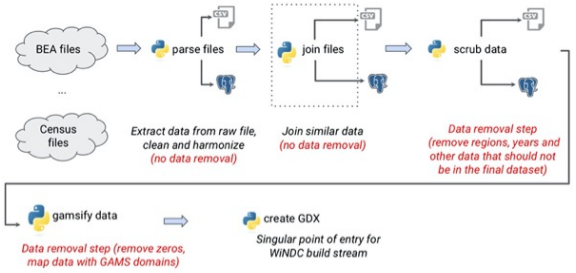
- Learning curve
- Under rapid development
  - Capabilities lacking
  - Frequent updates without backwards compatibility
- Syntax not as clean as GAMS[4]. No MPSGE[5] ..yet?

Benefits:

- SIIP compatibility
- Speed – especially when communicating between models
- Functionality: methods and structs
- One language to handle data processing and modeling

[6] J. Bezanson et al., SIAM Rev **59** (2017).
[5] T. F. Rutherford, Comput Econ **46** (1999).
[4] T. Brooke et al., (1998).

Building on the WiNDC Data Stream[7]



- Data operations stored in Julia "structs" and performed by general methods of SLiDE functions.
- User-friendly data operation definitions in human-readable YAML files to acquaint new users with Julia and encourage adoption.

[7] *Getting Started: WiNDC Data Stream*, WiNDC.

YAML-Defined Data Operations

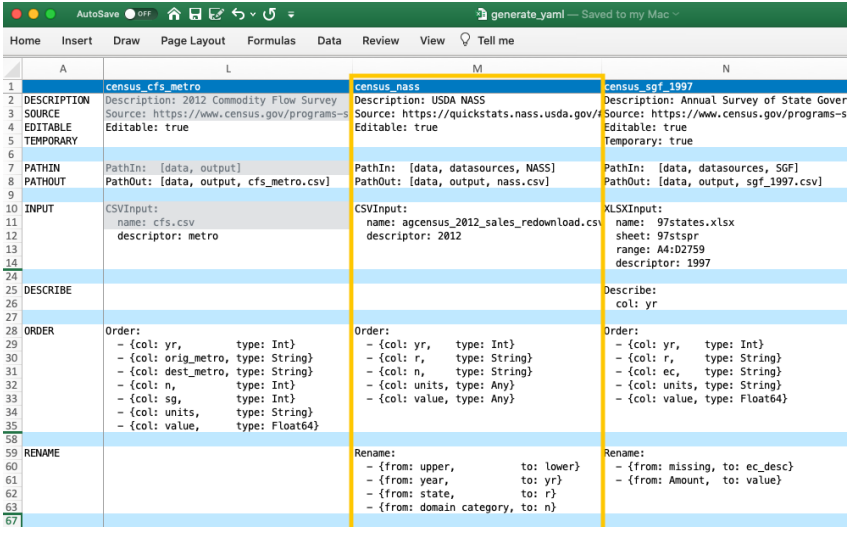
YAML definition:

```
PathIn:  [data, datasources, NASS]
PathOut: [data, output, nass.csv]

CSVInput:
  name: agcensus_2012_sales_redownload.csv
  descriptor: 2012

Order:
  - {col: yr,    type: Int}
  - {col: r,     type: String}
  - {col: n,     type: String}
  - {col: units, type: Any}
  - {col: value, type: Any}

Rename: # consistent with set names
  - {from: upper,           to: lower}
  - {from: year,            to: yr}
  - {from: state,           to: r}
  - {from: domain category, to: n}

Match: # for regular expressions
  on:     \((?<n>.*)\)
  input:  n
  output: n
```

Julia translation:

```
julia> y = read_file("PATH/TO/YAML/census_nass.yml")
Dict{Any,Any} with 11 entries:
  "PathIn"    => ["data", "datasources", "NASS"]
  "PathOut"   => ["data", "output", "nass.csv"]
  "CSVInput" => CSVInput(
    "agcensus_2012_sales_redownload.csv", "2012")
  "Drop"      => [Drop(:Program, "all", "=="),
                  Drop(:Period, "all", "=="), ...]
  "Rename"    => [Rename(:upper, :lower), ...]
  "Match"     => Match(r"\((?<n>.*)\)", :n, [:n])
  "Add"       => Add(:units, "us dollars (USD)")
  "Map"       => [Map("parse/regions.csv", [:from],
                      [:to], [:r], [:r], :left), ... ]
  "Replace"   => [Replace(:value, "(D)", 0), ...]
  "Operate"   => Operate(:col, "*", [:units],
                      [:units_factor], [:value, :factor↩
                          ], :value)
  "Order"     => Order([:yr, :r, :n, :units, :value],
    DataType[Int64, String, String, Any, Any])
```

Spreadsheet-summarized operations



	A	L	M	N
1		census_cfs_metro	census_nass	census_sgf_1997
2	DESCRIPTION	Description: 2012 Commodity Flow Survey	Description: USDA NASS	Description: Annual Survey of State Gover
3	SOURCE	Source: https://www.census.gov/programs-s	Source: https://quickstats.nass.usda.gov/#	Source: https://www.census.gov/programs-s
4	EDITABLE	Editable: true	Editable: true	Editable: true
5	TEMPORARY			Temporary: true
6				
7	PATHIN	PathIn: [data, output]	PathIn: [data, datasources, NASS]	PathIn: [data, datasources, SGF]
8	PATHOUT	PathOut: [data, output, cfs_metro.csv]	PathOut: [data, output, nass.csv]	PathOut: [data, output, sgf_1997.csv]
9				
10	INPUT	CSVInput:	CSVInput:	XLSXInput:
11		name: cfs.csv	name: agcensus_2012_sales_redownload.csv	name: 97states.xlsx
12		descriptor: metro	descriptor: 2012	sheet: 97stspr
13				range: A4:D2759
14				descriptor: 1997
24				
25	DESCRIBE			Describe:
26				col: yr
27				
28	ORDER	Order:	Order:	Order:
29		- {col: yr, type: Int}	- {col: yr, type: Int}	- {col: yr, type: Int}
30		- {col: orig_metro, type: String}	- {col: r, type: String}	- {col: r, type: String}
31		- {col: dest_metro, type: String}	- {col: n, type: String}	- {col: ec, type: String}
32		- {col: n, type: Int}	- {col: units, type: Any}	- {col: units, type: String}
33		- {col: sg, type: Int}	- {col: value, type: Any}	- {col: value, type: Float64}
34		- {col: units, type: String}		
35		- {col: value, type: Float64}		
58				
59	RENAME		Rename:	Rename:
60			- {from: upper, to: lower}	- {from: missing, to: ec_desc}
61			- {from: year, to: yr}	- {from: Amount, to: value}
62			- {from: state, to: r}	
63			- {from: domain category, to: n}	
67				

Example: National Agricultural Statistics Service (NASS)

Input:

Program	Year	Period	Week Ending	Geo Level	State	...	Domain Category	Value	CV (%)
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (111)	3,759,611,000	0.9
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (1111)	2,656,140,000	1.1
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (11111)	166,698,000	2.3
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (11112)	(D)	(D)

YAML definition:

```yaml
# Autogenerated from generate_yaml.xlsx
Description: USDA NASS
Source: https://quickstats.nass.usda.gov/#777837D3-E71B-323A-8137-D10646E77104
Editable: true

PathIn:  [data, datasources, NASS]
PathOut: [data, output, nass.csv]

CSVInput:
  name: agcensus_2012_sales_redownload.csv
  descriptor: 2012
```

Example: National Agricultural Statistics Service (NASS)

Before:

Program	Year	Period	Week Ending	Geo Level	State	...	Domain Category	Value	CV (%)
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (111)	3,759,611,000	0.9
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (1111)	2,656,140,000	1.1
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (11111)	166,698,000	2.3
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (11112)	(D)	(D)

YAML definition:

```
Drop:
  col: [Program, Period, Week Ending, Geo↩
        Level, State ANSI, Ag District,↩
        Ag District Code, County, ↩
       County ANSI, Zip Code, Region, ↩
       watershed_code, Watershed, ↩
       Commodity, Data Item, Domain, CV↩
        (%)]
  val: all
  operation: "=="
```

Example: National Agricultural Statistics Service (NASS)

Before:

Program	Year	Period	Week Ending	Geo Level	State	...	Domain Category	Value	CV (%)
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (111)	3,759,611,000	0.9
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (1111)	2,656,140,000	1.1
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (11111)	166,698,000	2.3
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (11112)	(D)	(D)

YAML definition:

```
Drop:
  col: [Program, Period, Week Ending, Geo↩
        Level, State ANSI, Ag District,↩
        Ag District Code, County, ↩
       County ANSI, Zip Code, Region, ↩
       watershed_code, Watershed, ↩
       Commodity, Data Item, Domain, CV↩
        (%)]
  val: all
  operation: "=="
```

Julia translation:

```
julia> y["Drop"]
17-element Array{Drop,1}:
 Drop(:Program, "all", "==")
 Drop(:Period, "all", "==")
 Drop(Symbol("Week Ending"), "all", "==")
 Drop(Symbol("Geo Level"), "all", "==")
 Drop(Symbol("State ANSI"), "all", "==")
 ⋮
 Drop(:Domain, "all", "==")
 Drop(Symbol("CV (%)"), "all", "==")

julia> df = edit_with(df, y["Drop"])
```

Example: National Agricultural Statistics Service (NASS)

After:

Year	State	Domain Category	Value
2012	WISCONSIN	NAICS CLASSIFICATION: (111)	3,759,611,000
2012	WISCONSIN	NAICS CLASSIFICATION: (1111)	2,656,140,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11111)	166,698,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11112)	(D)

YAML definition:

```
Drop:
  col: [Program, Period, Week Ending, Geo↩
        Level, State ANSI, Ag District,↩
        Ag District Code, County, ↩
       County ANSI, Zip Code, Region, ↩
       watershed_code, Watershed, ↩
       Commodity, Data Item, Domain, CV↩
        (%)]
  val: all
  operation: "=="
```

Julia translation:

```
julia> y["Drop"]
17-element Array{Drop,1}:
 Drop(:Program, "all", "==")
 Drop(:Period, "all", "==")
 Drop(Symbol("Week Ending"), "all", "==")
 Drop(Symbol("Geo Level"), "all", "==")
 Drop(Symbol("State ANSI"), "all", "==")
 ⋮
 Drop(:Domain, "all", "==")
 Drop(Symbol("CV (%)"), "all", "==")

julia> df = edit_with(df, y["Drop"])
```

Example: National Agricultural Statistics Service (NASS)

Before:

Year	State	Domain Category	Value
2012	WISCONSIN	NAICS CLASSIFICATION: (111)	3,759,611,000
2012	WISCONSIN	NAICS CLASSIFICATION: (1111)	2,656,140,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11111)	166,698,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11112)	(D)

YAML definition:

```
Rename: # consistent with set names
  - {from: upper,           to: lower}
  - {from: year,            to: yr}
  - {from: state,           to: r}
  - {from: domain category, to: n}
```

Example: National Agricultural Statistics Service (NASS)

Before:

Year	State	Domain Category	Value
2012	WISCONSIN	NAICS CLASSIFICATION: (111)	3,759,611,000
2012	WISCONSIN	NAICS CLASSIFICATION: (1111)	2,656,140,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11111)	166,698,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11112)	(D)

YAML definition:

```
Rename: # consistent with set names
  - {from: upper,           to: lower}
  - {from: year,            to: yr}
  - {from: state,           to: r}
  - {from: domain category, to: n}
```

Julia translation:

```
julia> y["Rename"]
4-element Array{Rename,1}:
 Rename(:upper, :lower)
 Rename(:year, :yr)
 Rename(:state, :r)
 Rename(Symbol("domain category"), :n)

julia> df = edit_with(df, y["Rename"])
```

Example: National Agricultural Statistics Service (NASS)

After:

yr	r	n	value
2012	WISCONSIN	NAICS CLASSIFICATION: (111)	3,759,611,000
2012	WISCONSIN	NAICS CLASSIFICATION: (1111)	2,656,140,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11111)	166,698,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11112)	(D)

YAML definition:

```
Rename: # consistent with set names
  - {from: upper,           to: lower}
  - {from: year,            to: yr}
  - {from: state,           to: r}
  - {from: domain category, to: n}
```

Julia translation:

```
julia> y["Rename"]
4-element Array{Rename,1}:
 Rename(:upper, :lower)
 Rename(:year, :yr)
 Rename(:state, :r)
 Rename(Symbol("domain category"), :n)

julia> df = edit_with(df, y["Rename"])
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value
2012	WISCONSIN	NAICS CLASSIFICATION: (111)	3,759,611,000
2012	WISCONSIN	NAICS CLASSIFICATION: (1111)	2,656,140,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11111)	166,698,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11112)	(D)

YAML definition:

```
Match: # for regular expressions
  on:     \((?<n>.*)\)
  input:  n
  output: n
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value
2012	WISCONSIN	NAICS CLASSIFICATION: (111)	3,759,611,000
2012	WISCONSIN	NAICS CLASSIFICATION: (1111)	2,656,140,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11111)	166,698,000
2012	WISCONSIN	NAICS CLASSIFICATION: (11112)	(D)

YAML definition:

```yaml
Match: # for regular expressions
  on:     \((?<n>.*)\)
  input:  n
  output: n
```

Julia translation:

```julia
julia> y["Match"]
Match(r"\((?<n>.*)\)", :n, [:n])

julia> df = edit_with(df, y["Match"])
```

Example: National Agricultural Statistics Service (NASS)

After:

yr	r	n	value
2012	WISCONSIN	111	3,759,611,000
2012	WISCONSIN	1111	2,656,140,000
2012	WISCONSIN	11111	166,698,000
2012	WISCONSIN	11112	(D)

YAML definition:

```
Match: # for regular expressions
  on:     \((?<n>.*)\)
  input:  n
  output: n
```

Julia translation:

```
julia> y["Match"]
Match(r"\((?<n>.*)\)", :n, [:n])

julia> df = edit_with(df, y["Match"])
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value	units
2012	WISCONSIN	111	3,759,611,000	us dollars (USD)
2012	WISCONSIN	1111	2,656,140,000	us dollars (USD)
2012	WISCONSIN	11111	166,698,000	us dollars (USD)
2012	WISCONSIN	11112	(D)	us dollars (USD)

YAML definition:

```yaml
Add:
  col: units
  val: us dollars (USD)
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value	units
2012	WISCONSIN	111	3,759,611,000	us dollars (USD)
2012	WISCONSIN	1111	2,656,140,000	us dollars (USD)
2012	WISCONSIN	11111	166,698,000	us dollars (USD)
2012	WISCONSIN	11112	(D)	us dollars (USD)

YAML definition:

```yaml
Add:
  col: units
  val: us dollars (USD)
```

Julia translation:

```
julia> y["Add"]
Add(:units, "us dollars (USD)")

julia> df = edit_with(df, y["Add"])
```

Example: National Agricultural Statistics Service (NASS)

After:

yr	r	n	value	units
2012	WISCONSIN	111	3,759,611,000	us dollars (USD)
2012	WISCONSIN	1111	2,656,140,000	us dollars (USD)
2012	WISCONSIN	11111	166,698,000	us dollars (USD)
2012	WISCONSIN	11112	(D)	us dollars (USD)

YAML definition:

```yaml
Add:
  col: units
  val: us dollars (USD)
```

Julia translation:

```
julia> y["Add"]
Add(:units, "us dollars (USD)")

julia> df = edit_with(df, y["Add"])
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value	units
2012	WISCONSIN	111	3,759,611,000	us dollars (USD)
2012	WISCONSIN	1111	2,656,140,000	us dollars (USD)
2012	WISCONSIN	11111	166,698,000	us dollars (USD)
2012	WISCONSIN	11112	(D)	us dollars (USD)

YAML definition:

```yaml
Map:
  - file:   [parse, regions.csv]
    from:   from
    to:     to
    input:  r
    output: r
  - file:   [parse, units.csv]
    from:   from
    to:     [to, factor, units_factor]
    input:  units
    output: [units, factor, units_factor]
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value	units
2012	WISCONSIN	111	3,759,611,000	us dollars (USD)
2012	WISCONSIN	1111	2,656,140,000	us dollars (USD)
2012	WISCONSIN	11111	166,698,000	us dollars (USD)
2012	WISCONSIN	11112	(D)	us dollars (USD)

YAML definition:

```
Map:
  - file:   [parse, regions.csv]
    from:   from
    to:     to
    input:  r
    output: r
  - file:   [parse, units.csv]
    from:   from
    to:     [to, factor, units_factor]
    input:  units
    output: [units, factor, units_factor]
```

Julia translation:

```
julia> y["Map"]
2-element Array{Map,1}:
 Map("parse/regions.csv", [:from], [:to], [:r], [:r], :left)
 Map("parse/units.csv", [:from:], [:to, :factor, :units_factor], [:units↩
       ], [:units, :factor, :units_factor], :left)

julia> df = edit_with(df, y["Map"])
```

Example: National Agricultural Statistics Service (NASS)

After:

yr	r	n	value	units	factor	units_factor
2012	wi	111	3,759,611,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	1111	2,656,140,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11111	166,698,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11112	(D)	us dollars (USD)	1.0e-6	millions of us dollars (USD)

YAML definition:

```
Map:
  - file:   [parse, regions.csv]
    from:   from
    to:     to
    input:  r
    output: r
  - file:   [parse, units.csv]
    from:   from
    to:     [to, factor, units_factor]
    input:  units
    output: [units, factor, units_factor]
```

Julia translation:

```
julia> y["Map"]
2-element Array{Map,1}:
 Map("parse/regions.csv", [:from], [:to], [:r], [:r], :left)
 Map("parse/units.csv", [:from:], [:to, :factor, :units_factor], [:units⟵
       ], [:units, :factor, :units_factor], :left)

julia> df = edit_with(df, y["Map"])
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value	units	factor	units_factor
2012	wi	111	3,759,611,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	1111	2,656,140,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11111	166,698,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11112	(D)	us dollars (USD)	1.0e-6	millions of us dollars (USD)

YAML definition:

```yaml
Replace:
  col:  value
  from: (D)
  to:   0
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value	units	factor	units_factor
2012	wi	111	3,759,611,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	1111	2,656,140,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11111	166,698,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11112	(D)	us dollars (USD)	1.0e-6	millions of us dollars (USD)

YAML definition:

```yaml
Replace:
  col:  value
  from: (D)
  to:   0
```

Julia translation:

```
julia> y["Replace"]
3-element Array{Replace,1}:
 Replace(:value, "(D)", 0)
 Replace(:value, "(H)", 0)
 Replace(:value, "(L)", 0)

julia> df = edit_with(df, y["Replace"])
```

Example: National Agricultural Statistics Service (NASS)

After:

yr	r	n	value	units	factor	units_factor
2012	wi	111	3,759,611,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	1111	2,656,140,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11111	166,698,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11112	0	us dollars (USD)	1.0e-6	millions of us dollars (USD)

YAML definition:

```yaml
Replace:
  col:  value
  from: (D)
  to:   0
```

Julia translation:

```
julia> y["Replace"]
3-element Array{Replace,1}:
 Replace(:value, "(D)", 0)
 Replace(:value, "(H)", 0)
 Replace(:value, "(L)", 0)

julia> df = edit_with(df, y["Replace"])
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value	units	factor	units_factor
2012	wi	111	3,759,611,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	1111	2,656,140,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11111	166,698,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11112	0	us dollars (USD)	1.0e-6	millions of us dollars (USD)

YAML definition:

```yaml
Operate:
  operation: "*"
  axis:      col
  from:      units
  to:        units_factor
  input:     [value, factor]
  output:    value
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value	units	factor	units_factor
2012	wi	111	3,759,611,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	1111	2,656,140,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11111	166,698,000	us dollars (USD)	1.0e-6	millions of us dollars (USD)
2012	wi	11112	0	us dollars (USD)	1.0e-6	millions of us dollars (USD)

YAML definition:

```yaml
Operate:
  operation: "*"
  axis:      col
  from:      units
  to:        units_factor
  input:     [value, factor]
  output:    value
```

Julia translation:

```
julia> y["Operate"]
Operate(:col, "*", [:units], [:units_factor], [:value, :factor], :value)

julia> df = edit_with(df, y["Operate"])
```

Example: National Agricultural Statistics Service (NASS)

After:

yr	r	n	value	units	value_0	factor	units_0
2012	wi	111	3759.611	millions of us dollars (USD)	3,759,611,000	1.0e-6	us dollars (USD)
2012	wi	1111	2656.14	millions of us dollars (USD)	2,656,140,000	1.0e-6	us dollars (USD)
2012	wi	11111	166.698	millions of us dollars (USD)	166,698,000	1.0e-6	us dollars (USD)
2012	wi	11112	0.0	millions of us dollars (USD)	0	1.0e-6	us dollars (USD)

YAML definition:

```yaml
Operate:
  operation: "*"
  axis:    col
  from:    units
  to:      units_factor
  input:   [value, factor]
  output: value
```

Julia translation:

```
julia> y["Operate"]
Operate(:col, "*", [:units], [:units_factor], [:value, :factor], :value)

julia> df = edit_with(df, y["Operate"])
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value	units	value_0	factor	units_0
2012	wi	111	3759.611	millions of us dollars (USD)	3,759,611,000	1.0e-6	us dollars (USD)
2012	wi	1111	2656.14	millions of us dollars (USD)	2,656,140,000	1.0e-6	us dollars (USD)
2012	wi	11111	166.698	millions of us dollars (USD)	166,698,000	1.0e-6	us dollars (USD)
2012	wi	11112	0.0	millions of us dollars (USD)	0	1.0e-6	us dollars (USD)

YAML definition:

```
Order:
  - {col: yr,    type: Int}
  - {col: r,     type: String}
  - {col: n,     type: String}
  - {col: units, type: Any}
  - {col: value, type: Any}
```

Example: National Agricultural Statistics Service (NASS)

Before:

yr	r	n	value	units	value_0	factor	units_0
2012	wi	111	3759.611	millions of us dollars (USD)	3,759,611,000	1.0e-6	us dollars (USD)
2012	wi	1111	2656.14	millions of us dollars (USD)	2,656,140,000	1.0e-6	us dollars (USD)
2012	wi	11111	166.698	millions of us dollars (USD)	166,698,000	1.0e-6	us dollars (USD)
2012	wi	11112	0.0	millions of us dollars (USD)	0	1.0e-6	us dollars (USD)

YAML definition:

```
Order:
  - {col: yr,    type: Int}
  - {col: r,     type: String}
  - {col: n,     type: String}
  - {col: units, type: Any}
  - {col: value, type: Any}
```

Julia translation:

```
julia> y["Order"]
Order([:yr, :r, :n, :units, :value],
    DataType[Int64, String, String, Any, Any])

julia> df = edit_with(df, y["Order"])
```

Example: National Agricultural Statistics Service (NASS)

After:

yr	r	n	units	value
2012	wi	111	millions of us dollars (USD)	3759.611
2012	wi	1111	millions of us dollars (USD)	2656.14
2012	wi	11111	millions of us dollars (USD)	166.698
2012	wi	11112	millions of us dollars (USD)	0.0

YAML definition:

```
Order:
  - {col: yr,    type: Int}
  - {col: r,     type: String}
  - {col: n,     type: String}
  - {col: units, type: Any}
  - {col: value, type: Any}
```

Julia translation:

```
julia> y["Order"]
Order([:yr, :r, :n, :units, :value],
    DataType[Int64, String, String, Any, Any])

julia> df = edit_with(df, y["Order"])
```

Example: National Agricultural Statistics Service (NASS)

In summary:

Input:

Program	Year	Period	Week Ending	Geo Level	State	...	Domain Category	Value	CV (%)
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (111)	3,759,611,000	0.9
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (1111)	2,656,140,000	1.1
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (11111)	166,698,000	2.3
CENSUS	2012	YEAR		STATE	WISCONSIN		NAICS CLASSIFICATION: (11112)	(D)	(D)

Output:

yr	r	n	units	value
2012	wi	111	millions of us dollars (USD)	3759.611
2012	wi	1111	millions of us dollars (USD)	2656.14
2012	wi	11111	millions of us dollars (USD)	166.698
2012	wi	11112	millions of us dollars (USD)	0.0

Maps are organized by purpose.

Parse. Standardize data across files for compatibility.
Ex: Regions, units, (unit conversion new to SLiDE)

Scale. Navigate between levels of the same type of data to refine focus.
Ex: core-based statistical area (CBSA)

Crosswalk. Navigate between different identifiers that represent identical or related information. This was challenging for sectoral information, with distinct codes for:

- Bureau of Economic Analysis (BEA)
- North American Industry Classification System (NAICS)
- Gross State Product (GSP)
- Standard Classification of Transported Goods (SCTG)
- WiNDC

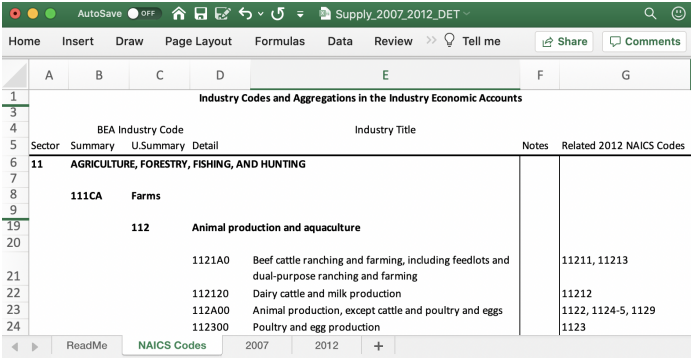
Sector Schema Crosswalks: Join on NAICS and WiNDC Codes

SCTG-to-NAICS[8]:

Table 2. Cross-reference between SCTG and NAICS

FAF Commodity	SCTG	NAICS
Live animals/fish	01	112
Cereal grains	02	111
Other ag prods.	03	111
Animal feed	04	311
Meat/seafood	05	311
Milled grain prods.	06	311
Other foodstuffs	07	311
Alcoholic beverages	08	312
Tobacco prods.	09	312
Building stone	10	212
Natural sands	11	212
Gravel	12	212
Nonmetallic minerals	13	212
Metallic ores	14	212
Coal	15	2121
Crude petroleum	16	211
Gasoline	17	324
Fuel oils	18	324

BEA-to-NAICS[9]:



[8] M. Anderson et al., Int J Traffic Transport Eng **2** (2013).

[9] *Input-Output Accounts Data*, BEA, (Oct. 2019).

Sector Crosswalk: Sample Output

Slice: Forestry, fishing, and related activities

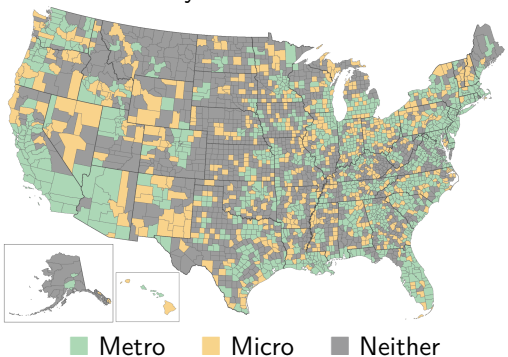
NAICS	BEA	GSP	SCTG	WiNDC Aggregate	WiNDC Sector	NAICS Description
1132				fof		Forest nurseries and gathering of forest products
1133				fof		Logging
1141				fof		Fishing
113	113000	5	25	fof	log_fof	Forestry and logging
114	114000	5		fof	fht_fof	Fishing, hunting and trapping
115	115000	5		fof	saf_fof	Support activities for agriculture and forestry
	113FF			fof		
112			1	fof		Animal production and aquaculture

Only two lines fully mapped:

NAICS	BEA	GSP	SCTG	WiNDC Aggregate	WiNDC Sector	NAICS Description
315	315000	28	30	alt	app_alt	Apparel manufacturing
113	113000	5	25	fof	log_fof	Forestry and logging

Regional Scaling

Counties by Area Status: 2017[10]



Current regional data availability:

Source	Region	State	County	CSA	CBSA
BEA					
Supply/Use	–	–	–	–	–
GSP	✓	✓	✓		✓
PCE	✓	✓	–	–	–
Census					
CFS	–	✓	–	✓	✓
NASS	–	✓	–	–	–
SGF	–	✓	–	–	–
UTD	–	✓	–	–	–
EIA					
Crude Oil	–	–	–	–	–
Emissions	–	✓	–	–	–
Heat Rate	–	–	–	–	–
SEDS	–	–	–	–	–

[10] *Counties by Metro/Micro Area Status: 2017*, Census Bureau, (2017).

Status

Development:

- Open-sourcing: https://github.com/NREL/SLiDE
- Code review(s)
- Readme/documentation

Features:

- Region scaling – up or down
- Model linkages – replace EIA data with ReEDS output
- KLEM nesting
- Household disaggregation

Outreach: SLiDE User Group (SLUG):

- Intern started yesterday
- Post-doc starting in the fall
- You?

References

[1] S. Cohen et al., *Regional Energy Deployment System (ReEDS) Model Documentation: Version 2018*, NREL/TP-6A20-72023 (National Renewable Energy Laboratory, Golden, CO, Apr. 2019).

[2] M. Muratori et al., "Future integrated mobility-energy systems: a modeling perspective", Renew Sustain Energ Rev **119**, 109541 (2020).

[3] T. Carson et al., "Exploring the Impacts of a National U.S. CO_2 Tax and Revenue Recycling Options with a Coupled Electricity-Economy Model", Clim Chang Econ **9**, 1840015 (2018).

[4] T. Brooke et al., "GAMS: A User's Guide", (1998).

[5] T. F. Rutherford, "Applied General Equilibrium Modeling with MPSGE as a GAMS Subsystem: An Overview of the Modeling Framework and Syntax", Comput Econ **46**, 1–46 (1999).

[6] J. Bezanson et al., "Julia: a fresh approach to numerical computing", SIAM Rev **59**, 65–98 (2017).

[7] *Getting Started: WiNDC Data Stream*, WiNDC, `https://windc.wisc.edu/datastream.html`.

[8] M. Anderson et al., "Validation of Disaggregate Methodologies for National Level Freight Data", Int J Traffic Transport Eng **2**, 51–54 (2013).

[9] *Input-Output Accounts Data*, BEA, (Oct. 2019) `https://www.bea.gov/industry/input-output-accounts-data`.

[10] *Counties by Metro/Micro Area Status: 2017*, Census Bureau, (2017) `https://www.census.gov/library/visualizations/2018/demo/2017-state-county-maps.html`.