Future PRIMAP data structures

Evaluation of available python libraries

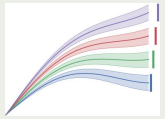
Overview: Contenders

Numfocus libraries

Built on pandas

SCMData

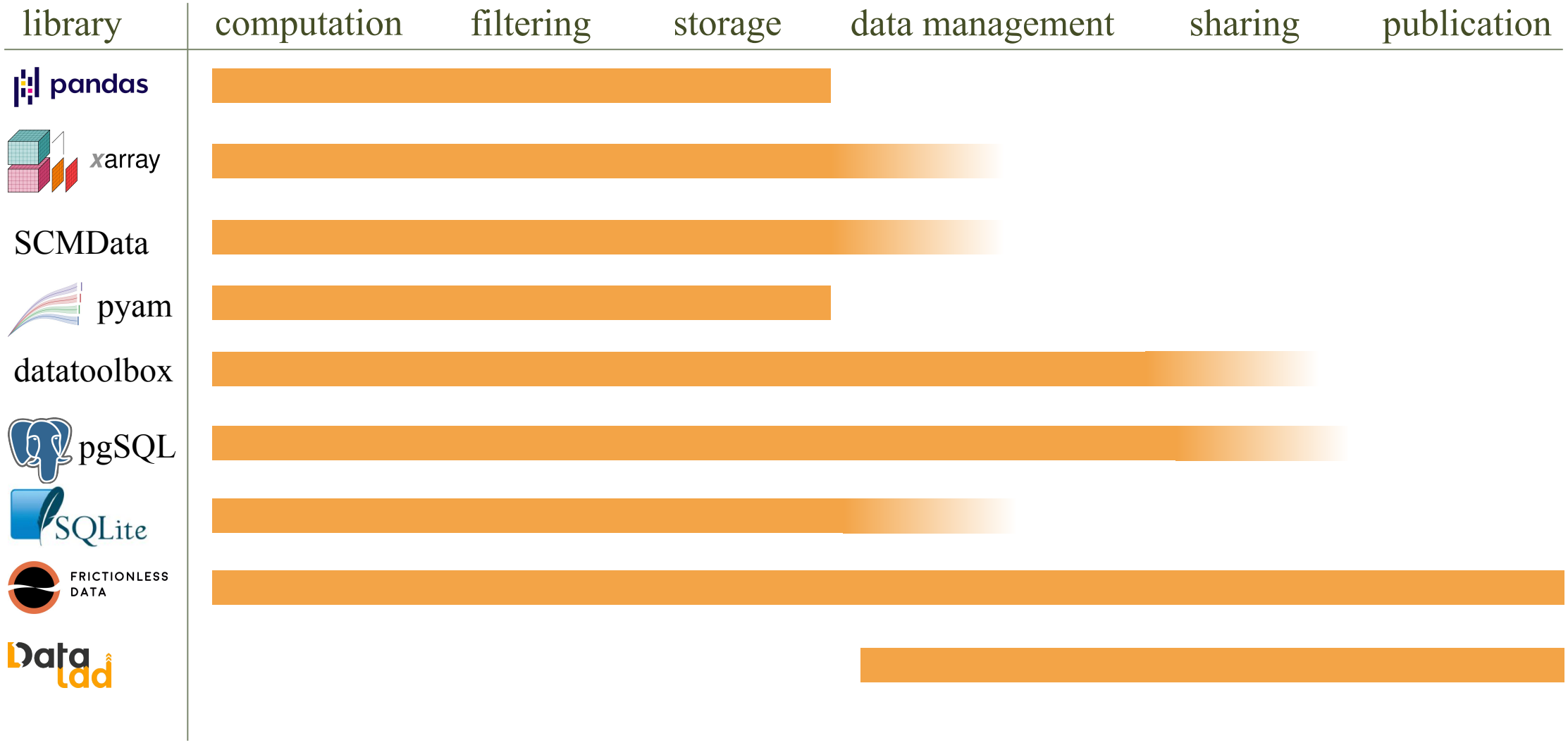
datatoolbox



pyam

pgSQL

Overview: Application Areas



Overview: Community, Institutional Support

library	
pandas	mature, institutional support, robust ecosystem
xarray	mature, institutional support, growing ecosystem
SCMData	in development, small dev team, responsive
pyam	established, small dev team
datatoolbox	in development, small dev team, close to pandas
pgSQL	mature, institutional support, robust ecosystem
SQLite	mature, institutional support, robust ecosystem, support pledged until at least 2050
FRICTIONLESS DATA	institutional support
Datalad	established, limited institutional support, medium dev team

Overview: Documentation

library	
pandas	★★★ extensive documentation, books, stackoverflow
xarray	★★★ extensive documentation, stackoverflow
SCMData	★★½ API documentation
pyam	★★ full documentation
datatoolbox	½ some tutorials
pgSQL	★★★ extensive documentation, books, stackoverflow
SQLite	★★★ extensive documentation, books, stackoverflow
FRICTIONLESS DATA	😵 Potemkin documentation: mission statements, intro videos, the rest 404 or wrong
Datalad	★★½ full documentation, ebook, videos

Computation: features and performance

library	aligned arithmetic	unit support	select	interpolate	resample
pandas	✓ 14s	✓	✓ 50ms	✓ 5ms	✓ 20ms
xarray	✓ 10ms	✓	✓ 5ms	✓ 100ms	✓ 300ms
SCMData	✘	✓	✓ 30ms	?	via pandas
pyam	✘	✓	✓ 24s	?	?
pgSQL	✓ 3ms	✓	✓ 20ms	✘	✓ 60ms
SQLite	✓ 5ms	✘	✓ 10ms	✘	✓ 60ms

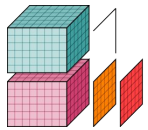
Selection syntax

pandas

```
a = (prm_emi
     .xs('IPC1', level='Category')
     .xs(slice('1900', '1990'), level='Date')
     .xs('CO2', level='Entity')
     .xs('HISTCR', level='Scenario'))
```

SCMData

```
a = prm_emi.filter(
    Category='IPC1',
    year=range(1900, 1991)
    Entity='CO2',
    Scenario='HISTCR')
```



xarray

```
a = prm_emi.loc[
    {'Category': ['IPC1'],
     'Date': slice('1900', '1990'),
     'Entity': ['CO2'],
     'Scenario': ['HISTCR']}]
```

pgSQL

```
create view a as
    select * from prm_emi
    where Category = 'IPC1' and
          Date < '1991-01-01'::date and
          Date >= '1900-01-01'::date and
          Entity = 'CO2' and
          Scenario = 'HISTCR'
```

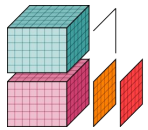
Aligned arithmetic syntax

pandas

```
(a + b).dropna()
```

SCMData

```
c = a.timeseries() + b.timeseries()
c = CustomScmRun(
    c.dropna('index', how='all')
     .dropna('columns', how='all'))
```



xarray

```
a + b
```

pgSQL

```
select a.emissions + b.emissions as results
       from a join b
       using (area, category, date, entity, scenario)
```

Memory + storage

library	RAM size	native storage size	lazy loading
pandas	200 MiB	60 MiB	✗
xarray	130 MiB	110 MiB	✓
SCMData	900 MiB	430 MiB	✗
pyam	500 MiB	194 MiB	✗
pgSQL	?	120 MiB	✓
SQLite	?	800 MiB	✓

Date handling

library	date < 1700	date < 1 AD
pandas	✗	✗
xarray	✓	✓
SCMData	✓	✗
pyam	✗	✗
pgSQL	✓	✓
SQLite	✓	✓

Data management, sharing, publication

Library	Metadata on table	Select on metadata	Select on data	share	publish
xarray	✓ On DataArray, Dataset	✗	✗	✗	✗
SCMData	✓ On SCMRun, Filesystem based DB	?	?	?	✗
pgSQL	✓ Using foreign keys	✓	✓	✓ Database server	✗
SQLite	✓ Using foreign keys	✓	✓	✗	✗
Datalad	✓ On Files, Datasets, aggregation	✓	✓	✓ Git	✓ Git

Select on metadata syntax

Task: select datasets which contain
data on the N2O emissions of Finland

pgSQL

```
select distinct primap_metadata_id, description, scenario, source
from primap_metadata join primap_data using (primap_metadata_id)
where primap_metadata.entity = 'N2O'
  and primap_data.area = 'FIN'
  and primap_metadata.category = 'IPC0'
```

Datalad

```
ds.search('annex.countries:FIN annex.entity:N2O annex.category:IPC0 type:file',
          mode='autofield')
```

Conclusion

- xarray + datalad
- pandas + own metadata handling + datalad
- pandas + postgresql

Pain Points:

- pandas: slow aligned arithmetic for large datasets
- pandas: unit handling beta quality
- xarray: sparse datasets unwieldy
- xarray: unit handling alpha quality
- postgresql: need server