Causal Probabilistic Programming with ChiRho

Jack Feser
Based on material by Sam Witty
Basis Research Institute

chirho

Public

Edit Pins | Unwatch 10 | Fork 6 | Starred 123

master | 102 Branches | 4 Tags | Go to file t | Add file | Code

djinnome Update dr_learner.ipynb (#506) ✓ 29ff860 · 5 days ago 192 Commits

.github/workflows	add dynamical to test requirements (#434)	last month
chirho	migrate soft conditioning and fix a few small explainable ...	last week
docs	Update dr_learner.ipynb (#506)	5 days ago
scripts	staging-causality (explainable reasoning as a separate m...	2 weeks ago
tests	migrate soft conditioning and fix a few small explainable ...	last week
.coveragerc	First commit	2 years ago
.gitignore	adding notebook tests (#308)	2 months ago
CONTRIBUTING.rst	Add CONTRIBUTING to sphinx docs (#425)	last month
LICENSE.md	Apache	2 years ago
Makefile	Added CONTRIBUTING.md file (#172)	6 months ago
README.rst	Update README.rst (#499)	last week
setup.cfg	Change name of package throughout codebase (#192)	6 months ago
setup.py	Staging branch for `chirho.robust` module development (...	2 weeks ago

README | Apache-2.0 license

About

An experimental language for causal reasoning

basisresearch.github.io/chirho/getting...

Readme
Apache-2.0 license
Activity
Custom properties
123 stars
10 watching
6 forks
Report repository

Releases 4

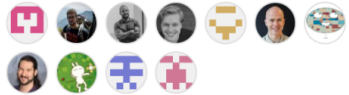
0.2.0 Latest
on Dec 11, 2023

+ 3 releases

Packages

No packages published
Publish your first package

Contributors 11



chirho Public

Edit Pins | Unwatch 10 | Fork 6 | Starred 123

master | 102 Branches | 4 Tags | Go to file t | Add file | Code

djinnome Update dr_learner.ipynb (#506) ✓ 29ff860 · 5 days ago 192 Commits

.github/workflows	add dynamical to test requirements (#434)	last month
chirho	migrate soft conditioning and fix a few small explainable ...	last week
docs	Update dr_learner.ipynb (#506)	5 days ago
scripts	staging-causality (explainable reasoning as a separate m...	2 weeks ago
tests	migrate soft conditioning and fix a few small explainable ...	last week
.coveragerc	First commit	2 years ago
.gitignore	adding notebook tests (#308)	2 months ago
CONTRIBUTING.rst	Add CONTRIBUTING to sphinx docs (#425)	last month
LICENSE.md	Apache	2 years ago
Makefile	Added CONTRIBUTING.md file (#172)	6 months ago
README.rst	Update README.rst (#499)	last week
setup.cfg	Change name of package throughout codebase (#192)	6 months ago
setup.py	Staging branch for `chirho.robust` module development (...	2 weeks ago

README | Apache-2.0 license

Test passing

About

An experimental language for causal reasoning

basisresearch.github.io/chirho/getting...

Readme
Apache-2.0 license
Activity
Custom properties
123 stars
10 watching
6 forks
Report repository

Releases 4

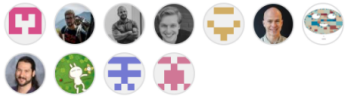
0.2.0 Latest
on Dec 11, 2023

+ 3 releases

Packages

No packages published
Publish your first package

Contributors 11



Motivation: Cause and effect in Biology

Scientific questions are often fundamentally causal in nature.

Motivation: Cause and effect in Biology

Scientific questions are often fundamentally causal in nature.

A biologist may ask:

- “What cancer drug candidates ***cause*** tumor cell death or block metastasis?”

Motivation: Cause and effect in Biology

Scientific questions are often fundamentally causal in nature.

A biologist may ask:

- “What cancer drug candidates ***cause*** tumor cell death or block metastasis?”
- “Would the patient have survived **had they received** the cancer drug A?”

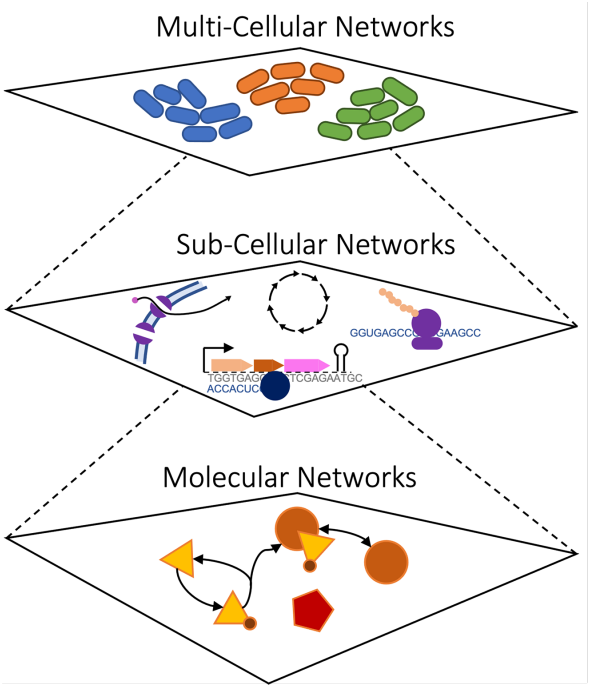
Motivation: Cause and effect in Biology

Scientific questions are often fundamentally causal in nature.

A biologist may ask:

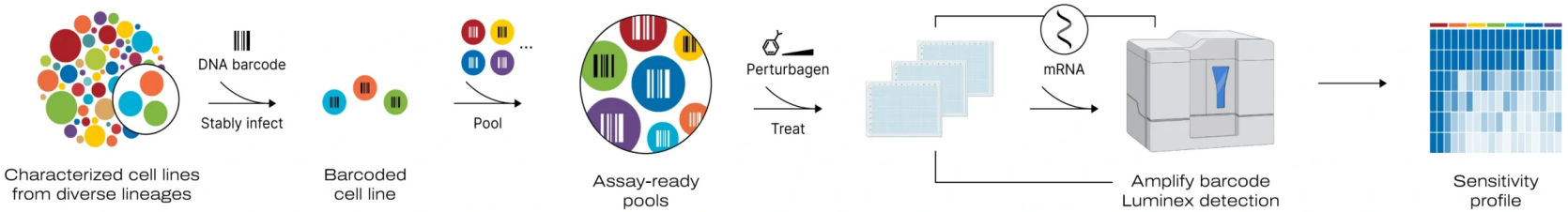
- “What cancer drug candidates ***cause*** tumor cell death or block metastasis?”
- “Would the patient have survived **had they received** the cancer drug A?”
- “**Why** did drug A have the effect that it did?”

Challenge: Biology is complicated



(source: vivariumlab.org)

Challenge: Biological data is complicated



(source: theprismlab.org)

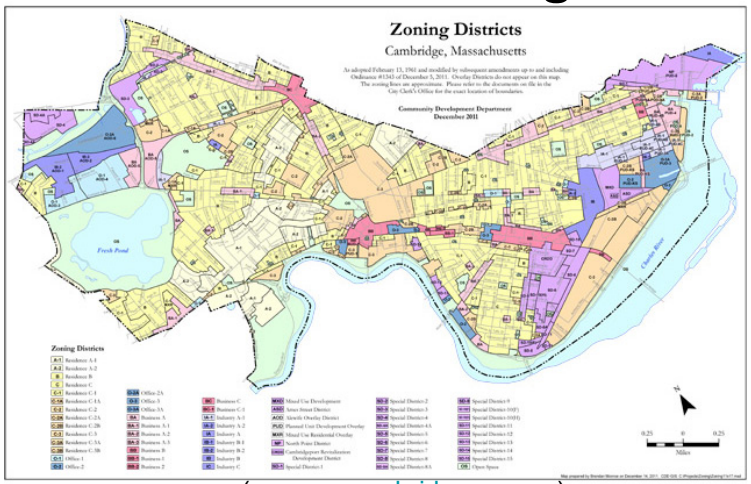
Public Health

(source: commonwealthfund.org)

Astronomy

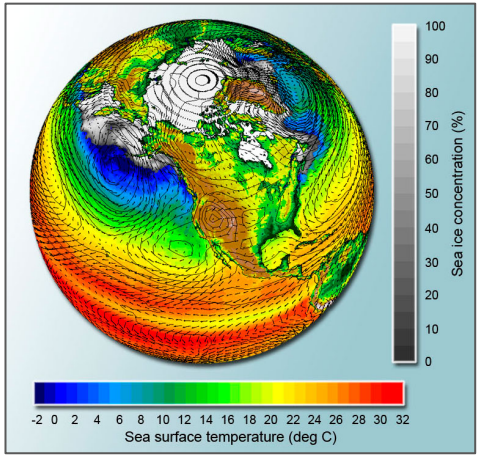
(source: nsf.gov)

Urban Planning



(source: cambridgema.gov)

Climate Science



(source: nsf.gov)

ChiRho turns causal inference into programming.

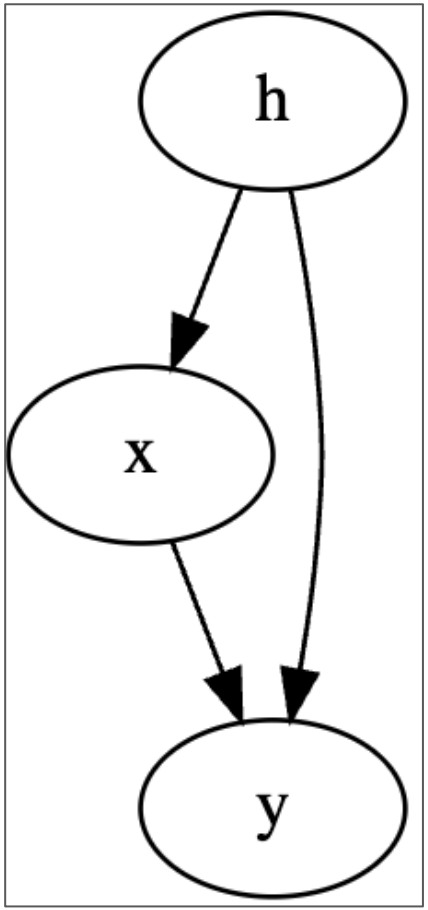
ChiRho turns causal inference into probabilistic programming.

Basic Usage

```python
import torch
import pyro
import pyro.distributions as dist
from chirho.interventional.handlers import do

pyro.set_rng_seed(101)

# Define a causal model with single confounder h
def model():
    h = pyro.sample("h", dist.Normal(0, 1))
    x = pyro.sample("x", dist.Normal(h, 1))
    y = pyro.sample("y", dist.Normal(x + h, 1))
    return y

# Define a causal query (here intervening on x)
def queried_model():
    return do(model, {"x": 1})

# Generate 10,000 samples from the observational distribution P(y) ~ N(0, 2)
obs_samples = pyro.infer.Predictive(model, num_samples=1000)()["y"]

# Generate 10,000 samples from the interventional distribution P(y | do(X=1))
int_samples = pyro.infer.Predictive(queried_model(), num_samples=1000)()["y"]
```

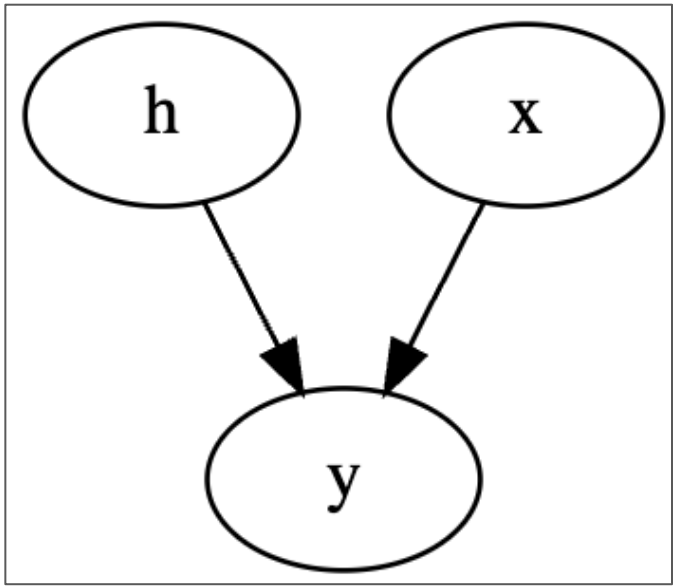
Basic Usage

```python
import torch
import pyro
import pyro.distributions as dist
from chirho.interventional.handlers import do

pyro.set_rng_seed(101)

# Define a causal model with single confounder h
def model():
    h = pyro.sample("h", dist.Normal(0, 1))
    x = pyro.sample("x", dist.Normal(h, 1))
    y = pyro.sample("y", dist.Normal(x + h, 1))
    return y

# Define a causal query (here intervening on x)
def queried_model():
    return do(model, {"x": 1})

# Generate 10,000 samples from the observational distribution P(y) ~ N(0, 2)
obs_samples = pyro.infer.Predictive(model, num_samples=1000)()["y"]

# Generate 10,000 samples from the interventional distribution P(y | do(X=1))
int_samples = pyro.infer.Predictive(queried_model(), num_samples=1000)()["y"]
```



Basic Usage

```python
import torch
import pyro
import pyro.distributions as dist
from chirho.interventional.handlers import do

pyro.set_rng_seed(101)

# Define a causal model with single confounder h
def model():
    h = pyro.sample("h", dist.Normal(0, 1))
    x = pyro.sample("x", dist.Normal(h, 1))
    y = pyro.sample("y", dist.Normal(x + h, 1))
    return y

# Define a causal query (here intervening on x)
def queried_model():
    return do(model, {"x": 1})

# Generate 10,000 samples from the observational distribution P(y) ~ N(0, 2)
obs_samples = pyro.infer.Predictive(model, num_samples=1000)()["y"]

# Generate 10,000 samples from the interventional distribution P(y | do(X=1))
int_samples = pyro.infer.Predictive(queried_model(), num_samples=1000)()["y"]
```



GO TO TUTORIAL

Question: “What is the direct effect of ***treatment*** on an ***outcome***, ignoring paths through particular ***mediators.***”

Question: “What is the direct effect of ***treatment*** on an ***outcome***, ignoring paths through particular ***mediators.***”

Translation: “What would ***outcome*** be if we intervene on ***treatment,*** and intervene on ***mediators*** to be what they would have been had we not intervened on ***treatment***?”

Direct and Indirect Effects. Pearl 2001

Advanced Usage - Mediation Analysis

```
class NaturalDirectEffectModel(pyro.nn.PyroModule):

    def __init__(self, causal_model: MediationModel):
        super().__init__()
        self.causal_model = causal_model

    @pyro.infer.config_enumerate
    def forward(self, x, x_prime):
        with MultiWorldCounterfactual(), \
                do(actions=dict(fam_int=(x, x_prime))), \
                do(actions=dict(sub_exp=lambda Z_: gather(Z_, IndexSet(fam_int={2})))), \
                pyro.plate("data", size=x.shape[0], dim=-1):

            ys = self.causal_model()
            ys_xprime = gather(ys, IndexSet(fam_int={2}, sub_exp={0}))  # y_x'
            ys_x = gather(ys, IndexSet(fam_int={1}, sub_exp={1}))  # y_x,z
            return ys_xprime - ys_x
```

Advanced Usage - Mediation Analysis

```
class NaturalDirectEffectModel(pyro.nn.PyroModule):

    def __init__(self, causal_model: MediationModel):
        super().__init__()
        self.causal_model = causal_model

    @pyro.infer.config_enumerate
    def forward(self, x, x_prime):
        with MultiWorldCounterfactual(), \
                do(actions=dict(fam_int=(x, x_prime))), \
                do(actions=dict(sub_exp=lambda Z_: gather(Z_, IndexSet(fam_int={2})))), \
                pyro.plate("data", size=x.shape[0], dim=-1):

            ys = self.causal_model()
            ys_xprime = gather(ys, IndexSet(fam_int={2}, sub_exp={0}))  # y_x'
            ys_x = gather(ys, IndexSet(fam_int={1}, sub_exp={1}))  # y_x,z
            return ys_xprime - ys_x
```

world-splitting

Advanced Usage - Mediation Analysis

```
class NaturalDirectEffectModel(pyro.nn.PyroModule):

    def __init__(self, causal_model: MediationModel):
        super().__init__()
        self.causal_model = causal_model

    @pyro.infer.config_enumerate
    def forward(self, x, x_prime):
        with MultiWorldCounterfactual(), \
                do(actions=dict(fam_int=(x, x_prime))), \
                do(actions=dict(sub_exp=lambda Z_: gather(Z_, IndexSet(fam_int={2})))), \
                pyro.plate("data", size=x.shape[0], dim=-1):

            ys = self.causal_model()
            ys_xprime = gather(ys, IndexSet(fam_int={2}, sub_exp={0}))  # y_x'
            ys_x = gather(ys, IndexSet(fam_int={1}, sub_exp={1}))  # y_x,z
            return ys_xprime - ys_x
```

intervention

ChiRho’s Design Goals

Expressive Models

ChiRho's Design Goals

Expressive Models

Expressive Questions

ChiRho's Design Goals

Expressive Models

Modular

Expressive Questions

ChiRho's Design Goals

Expressive Models

Modular

Extensible

Expressive Questions

ChiRho's Design - Built on Pyro

Expressive Models

Modular

Extensible

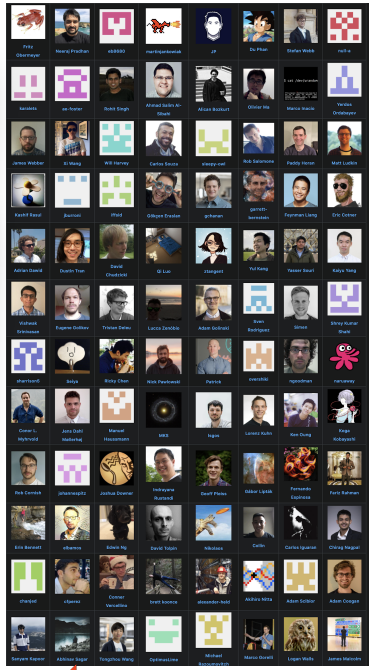
Expressive Questions

ChiRho's Design - Built on Pyro

Used by machine learners

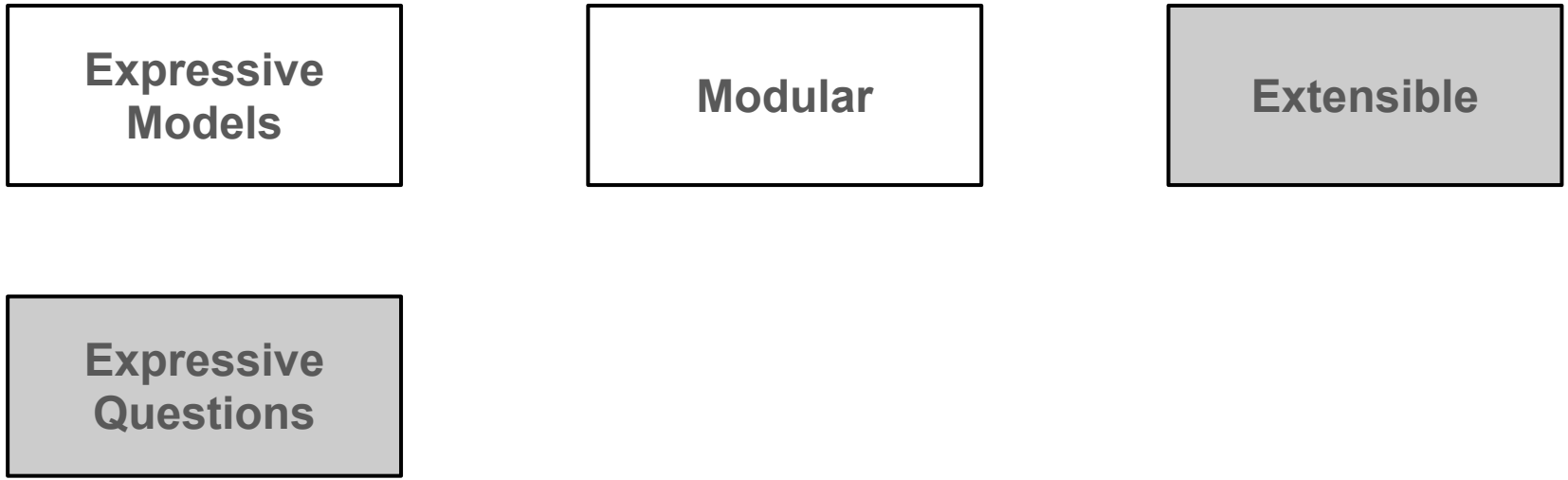
Used by scientists

Built by a community



- Released at Uber in late 2017, moved to Linux Foundation in early 2019
- Built on Meta's PyTorch and Google's JAX deep learning frameworks
- 10k+ GitHub stars, 150+ contributors, 1100+ citations, 750k+ downloads/month
- Core development team members at Broad Institute, Google, Generate, Basis

ChiRho's Design - Composable Effect Handlers



Operations

- Core primitives who’s interpretation can be modified by **effect handlers**.
- For example, the ***intervene*** operation replaces an observed value with an intervened value during execution by default.

Effect Handlers

- Higher order functions that modify the behavior of some **operations**.
- For example, the ***MultiWorldCounterfactual*** effect handler changes the behavior of ***intervene*** to return both observed and intervened values simultaneously.

Demo - Dynamical Systems for Epidemiology

https://github.com/BasisResearch/chirho/blob/master/docs/source/dynamical_intro.ipynb

Acknowledgements

Funding:

- CZI EOSS & Single Cell Data Insights
- DARPA ASKEM
- Survival and Flourishing Fund

People:

- Basis: **Eli Bingham, Raj Agrawal, Andy Zane**
- PNNL: Jeremy Zucker
- ChiRho and Pyro contributors
- ASKEM collaborators at PNNL and UT-Austin

Install, learn, discuss & contribute:
github.com/BasisResearch/ChiRho

Contact: sam@basis.ai,
zenna@basis.ai

About Basis:
basis.ai/about

Join us:
basis.ai/join-us