Infergo — http://inference.org/

**Models are written in Go.**
- Relies on automatic differentiation for inference.
- Works anywhere where Go does.
- No external dependencies.
- Licensed under the MIT license.

**Why Go?**
- Comes with parser and type checker.
- Compiles and runs fast.
- Allows efficient parallel execution, via goroutines.
- Popular for server-side programming.

**Automatic differentiation**
- Reverse-mode autodiff via source code transformation.
- Automatic selective differentiation of models.

**Case studies**

**& schools**

**Infergo**

```go
var breastCancer = flip(0.61)
var benignCyst = flip(0.3)

// positiveHamogram =
// (breastCancer & flip(0.5))
// (benignCyst & flip(0.5))
condition(positiveHamogram)

return breastCancer
```

**Rules**

- One language for system and model
- Common data structures
- Inference code re-used in simulation

**Implementation**

- Infergo — http://bitbucket.org/dtolpin/infergo
- Infergo — http://bitbucket.org/dtolpin/infergo/studies
- GoGP, a Gaussian process library — http://bitbucket.org/dtolpin/infergo-studies

**Performance**

- Model: breastCancer, benignCyst
- Compilation: 50 ms
- Execution: 60 ms

<table>
<thead>
<tr>
<th>Performance</th>
<th>model</th>
<th>compilation</th>
<th>execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Infergo</td>
<td>0.50</td>
<td>50</td>
<td>0.60</td>
</tr>
<tr>
<td>Turing</td>
<td>2.50</td>
<td>12</td>
<td>2.0</td>
</tr>
<tr>
<td>Stan</td>
<td>1.20</td>
<td>1.20</td>
<td>1.20</td>
</tr>
</tbody>
</table>

**Platforms**

- Linux
- Go
- Windows

**Data**

- Data on breast cancer patients
- Data on chest x-rays

**Challenges**

- Inferentiable programming
- Simulation vs. inference
- Data
- Deployment

**Languages**

- WebPPL
- Infer.NET
- Gen
- Church
- Brich
- BLOG
- Anglican

**Languages**

- PPLs participated in PROBPROG 2018:
  - 45 PPLs listed on Wikipedia
  - 18 PPLs participated in PROBPROG 2018

To name a few:

**Inference**

- Must be run backwards.
- May contain conditions, loops, recursions.

**Streaming & stochasticity**

```go
func (s *StreamModel) Observe(x []float64) float64 {
    ll := Normal.Logpdf(0, 1, x...)
    return ll + ll...
}
```

**Model composition**

```go
func (m *Model) Observe(x []float64) float64 {
    ll := Normal.Logpdf(0, 1, x...)
    return ll + ll...
}
```

**Automatic differentiation**

```go
ll := Normal.Logps(x[0], m.Sigma[i], y)
return ll + ll...
```

**Why Go?**

- Comes with parser and type checker.
- Compiles and runs fast.
- Allows efficient parallel execution, via goroutines.
- Popular for server-side programming.

**Languages**

- PPLs participated in PROBPROG 2018:
  - 45 PPLs listed on Wikipedia
  - 18 PPLs participated in PROBPROG 2018

To name a few:

**Inference**

- Must be run backwards.
- May contain conditions, loops, recursions.

**Languages**

- PPLs participated in PROBPROG 2018:
  - 45 PPLs listed on Wikipedia
  - 18 PPLs participated in PROBPROG 2018

To name a few: