Overview

- Last time: Nonmonotonic logic, Bayes nets, graphical models, ...

- Today:
  - Stochastic lambda calculus
  - Probabilistic First-Order Logic
Part I:
Stochastic Lambda Calculus
Probabilistic programming languages give rise to more general formalisms for defining probability distributions, with graphical models as a basic special case.

Stochastic Lambda Calculus: a perspicuous, simple formalism.
The $\lambda$-terms are given by a set of variables $\mathcal{V}$ and defined inductively:

- Each variable $x \in \mathcal{V}$ is a $\lambda$-term.
- If $M$ and $N$ are $\lambda$-terms, then so is $M(N)$.
- If $x \in \mathcal{V}$ and $M$ is a $\lambda$-term, then $\lambda x.M$ is a $\lambda$-term.

Main reduction rule, $\beta$-conversion:

$$(\lambda x.M)(N) \rightarrow_{\beta} M[N/x]$$

Let $\rightarrow^{*}_{\beta}$ be the reflexive, transitive closure of $\rightarrow_{\beta}$. 
Church-Rosser for $\rightarrow^\ast_\beta$
Church-Rosser for $\rightarrow^*_{\beta}$
Corollary
If a $\lambda$-term has a $\beta$-normal form, it is unique.
Boolean Logic in Lambda Calculus

\[ \top \equiv \lambda x.\lambda y.x \quad \bot \equiv \lambda x.\lambda y.y \]

\[ \text{and} \equiv \lambda a.\lambda b.\, b\, a\, b \quad \text{if}_\text{then}_\text{else} \equiv \lambda a.\lambda b.\lambda c.\, a\, b\, c \]

\[ \text{not} \equiv \lambda a.\, a\, \bot\top \]

For \( M \) a boolean function of \( v_1, \ldots, v_n \), and \( T_1, \ldots, T_n \in \{ \top, \bot \} \):

\[
(\lambda \nu_1 \ldots \lambda \nu_n.M)(T_1)\ldots(T_n) \stackrel{\beta}{\rightarrow}^* \begin{cases} \top \\ \bot \end{cases}
\]

depending on whether the boolean formula is true or false with this input.
Numbers in Lambda Calculus

\[ 0 \equiv \lambda f.\lambda x.x \]
\[ 1 \equiv \lambda f.\lambda x.f(x) \]
\[ 2 \equiv \lambda f.\lambda x.f(f(x)) \]
\[ \vdots \]
\[ n \equiv \lambda f.\lambda x.f^n(x) \]
\[ \text{succ} \equiv \lambda n.\lambda f.\lambda x.f(n(f(x))) \]

And so on …
Fixed-Point Combinators

\[ \Theta \equiv (\lambda x.\lambda y.y(xxy))(\lambda x.\lambda y.y(xxy)) \]

‘Turing’s combinator’ \( \Theta \) has the feature that, for any term \( M \),

\[ \Theta M \xrightarrow{\beta}^* M(\Theta M) . \]

This allows great flexibility, e.g., in defining recursive functions. But it also means trouble for logic, viz. Curry’s Paradox.
Curry’s Paradox

Suppose we devise a proof system in \( \lambda \)-calculus with the following rules:

\[
\begin{align*}
A \supset B & \quad \frac{A}{B} \\
(A \supset (A \supset B)) \supset (A \supset B) & \\
A & \quad \frac{A}{A = \beta B}
\end{align*}
\]

Define \( M \equiv \lambda x. x \supset (x \supset \bot) \), and let \( N \equiv \Theta M \). It is easy to show

\[ N = \beta N \supset (N \supset \bot) . \]

But then we can evidently prove \( \bot \):

1. \( (N \supset (N \supset \bot)) \supset (N \supset \bot) \)
2. \( N \supset (N \supset \bot) \)
3. \( N \)
4. \( N \supset \bot \)
5. \( \bot \)
Adding Probability

Stochastic lambda calculus adds to lambda calculus a new operator:

\[ M \oplus N \]

with intended interpretation of probabilistic choice between \( M \) and \( N \).

Write \( M \overset{p}{\rightarrow} N \) for: \( M \) reduces to normal form \( N \) with probability \( p \).

In particular, then, \( M \oplus N \overset{\frac{1}{2}}{\rightarrow} M \) and \( M \oplus N \overset{\frac{1}{2}}{\rightarrow} N \) if \( M \) and \( N \) are in normal form, and more generally,

\[ \sum_{M \overset{p}{\rightarrow} N} p \leq 1 \]

Stochastic \( \lambda \)-terms are themselves random variables!
Evidently, this allows us to define term denoting a 50/50 ‘coin flip’:

$$flip \equiv \top \oplus \bot$$

What about other types of coin flips, i.e., Bernoulli variables?
**Theorem (von Neumann)**

Suppose $p \in [0, 1]$ is computable, i.e., there is a $\lambda$-term $N$ such that

$$N(k) \rightarrow^* \begin{cases} \top & \text{if the } k^{\text{th}} \text{ digit in the binary expansion of } p \text{ is } 1 \\ \bot & \text{if the } k^{\text{th}} \text{ digit is } 0 \end{cases}$$

Then there is a stochastic $\lambda$-term $M$ such that: $M \xrightarrow{p} \top$ and $M \xrightarrow{1-p} \bot$.

**Proof.**

Define $E$ to be the following term:

$$E \equiv \lambda e. \lambda k. \text{flip} \left( N(k) \ e(\text{succ}(k)) \ \bot \right) \left( N(k) \ \top \ e(\text{succ}(k)) \right).$$

and let $M \equiv \Theta E$, which returns $\top$ with probability $p$ and $\bot$ with $1 - p$. 
Theorem (von Neumann)
The same result holds for arbitrary domains, not just $\top$ and $\bot$. 
Defining Graphical Models

For exposition, assume we are working with binary variables, and let $T_p$ be a term that returns $\top$ with probably $p$, and $\bot$ with probability $1 - p$. 
Defining Graphical Models

\[ P(X) = 0.1 \]
\[ P(Y) = 0.2 \]

\[ P(Z|X, Y) = 0.9 \]
\[ P(Z|X, \overline{Y}) = 0.8 \]
\[ P(Z|\overline{X}, Y) = 0.6 \]
\[ P(Z|\overline{X}, \overline{Y}) = 0.2 \]
Defining Graphical Models

\[
\begin{align*}
X & := T_{0.1} \\
Y & := T_{0.2} \\
Z & := \text{if } (X \text{ and } Y) \text{ then } T_{0.9} \\
& \quad \text{elif } (X \text{ and not } Y) \text{ then } T_{0.8} \\
& \quad \text{elif } (\text{not } X \text{ and } Y) \text{ then } T_{0.6} \\
& \quad \text{elif } (\text{not } X \text{ and not } Y) \text{ then } T_{0.2}
\end{align*}
\]
PCFGs

\[
S \ := \ \langle NP, VP \rangle \\
VP \ := \ \text{if } T_{0.75} \text{ then } \langle V, NP \rangle \text{ else } \langle V, NP, PP \rangle \\
NP \ := \ \text{if } T_{0.7} \text{ then } \langle \text{det}, N \rangle \text{ else } \langle NP, PP \rangle \\
PP \ := \ \langle P, NP \rangle \\
\vdots \\
N \ := \ \text{if } T_{0.02} \text{ then } \text{cat else } \ldots
\]
Given (meta-)variable names for programs,

\[ X_1, \ldots, X_n, \]

which we can think of as random variables, we can ask questions such as:

\[ P(X_i = v) \]
\[ P(X_i = v | Y_1 = u_1, \ldots, Y_m = u_m) \]

\[ \ldots \]

for an extremely general class of random variables.

In principle one could import the same tools on probability-preserving deduction—of Suppes, Adams, etc., from Lecture 1—to this setting. But one must be careful, witness Curry’s Paradox, etc.

(N.B. For more on actual programming languages based on these ideas, viz. Church, etc., see Noah Goodman’s class next week.)
Part II: Probabilistic First-Order Logic
Let $\mathcal{L}$ be a first-order logical language, given by:

- a set $\mathcal{C}$ of individual constants;
- a set $\mathcal{V}$ of individual variables,
- a set $\mathcal{P}$ of predicate variables.

Terms and formulas of $\mathcal{L}$ are defined as usual:

$$\varphi ::= R(t_1, \ldots, t_n) \mid \varphi \land \varphi \mid \neg \varphi \mid \exists x \varphi \mid \forall x \varphi$$
Define $S_L$ to be the set of sentences of $L$, i.e., formulas with no free variables, and $S^0_L$ to be the set of quantifier-free sentences of $L$.

As before, a probability on $L' \subseteq S_L$ is a function $P : L' \rightarrow [0, 1]$, with

- $P(\varphi) = 1$, for any first-order tautology $\varphi$;
- $P(\varphi \lor \psi) = P(\varphi) + P(\psi)$, whenever $\models \neg(\varphi \land \psi)$.

Question: Given a probability $P : S^0_L \rightarrow [0, 1]$, is there a natural extension of $P$ to all of $S_L$, including quantified sentences?

(Cf. Markov Logic.)
Question: Given a probability $P : S_L^0 \rightarrow [0, 1]$, is there a natural extension of $P$ to all of $S_L$, including quantified sentences?

If there are only finitely many constants $c$ such that $P(R(c)) > 0$, then:

$$P(\exists x R(x)) = P(\bigvee R(c))$$

What about in the case where the number of $c$ is infinite?
Example

Consider a simple first-order arithmetical language $\mathcal{L}$, with a constant $n$ for each $n \in \mathbb{N}^+$. Let $R(x)$ be a one-place predicate. Define a probability function $P : S_\mathcal{L}^0 \rightarrow [0, 1]$ on the quantifier-free sentences as follows:

$\begin{align*}
\rightarrow P(R(n)) &= 1/2^{n+1}, \text{ for all } n \in \mathbb{N}; \\
\rightarrow P(\bigwedge_{i \leq k} R(n_i)) &= \prod_{i \leq k} P(R(n_i)).
\end{align*}$

Then, intuitively,

$P(\exists x P(x)) = \sum_{n=2}^{\infty} \frac{1}{2^n} = \frac{1}{2}.$
Definition (Gaifman’s Condition)

A probability $P^* : S_L \rightarrow [0, 1]$ satisfies the Gaifman condition if for all formulas with one free variable $\varphi(x)$:

$$P^*(\exists x \varphi(x)) = \sup \left\{ P^*\left( \bigvee_{i=1}^{n} \varphi(c_i) \right) \mid c_1, \ldots, c_n \in C \right\} ,$$

or equivalently,

$$P^*(\forall x \varphi(x)) = \inf \left\{ P^*\left( \bigwedge_{i=1}^{n} \varphi(c_i) \right) \mid c_1, \ldots, c_n \in C \right\} .$$

Theorem (Gaifman 1964)

Given $P : S_L^0 \rightarrow [0, 1]$, there is exactly one extension $P^*$ of $P$ to all of $S_L$ that satisfies the Gaifman condition.
So far we have been focusing on probability functions $P$ defined on a language $\mathcal{L}$. But we can also start with a probability measure $\mu$ over models and induce a probability on formulas.

Recall models of first-order languages:

$$\mathcal{M} = \langle D, I, \gamma \rangle$$

- $D$ is a set of individuals
- $I$ is a function interpreting predicates, constants, etc.
- $\gamma$ is an assignment function for variables.
Consider a set of models $\mathcal{M} = \bigcup_{i \in I} \{ \mathcal{M}_i \}$, and $\sigma$-algebra $\mathcal{E}$ on $\mathcal{M}$. We will require that each set 

$$M_\varphi := \{ \mathcal{M} \in \mathcal{M} : \mathcal{M} \models \varphi \},$$ 

for $\varphi \in S_L$ is measurable, i.e., an element of $\mathcal{E}$.

We can define a measure $\mu : \mathcal{E} \to [0, 1]$, which induces an obvious probability $P_\mu : S_L \to [0, 1]$ on our first-order language $L$:

$$P_\mu(\varphi) = \mu(M_\varphi).$$

It is then easy to show that $P_\mu$ is a probability in our previous sense. Question: Does $P_\mu$ always satisfy the Gaifman condition?
Suppose that every element of every domain of every model is named by a constant, according to all interpretation functions. Then:

Fact

Given μ, the probability $P_\mu$ satisfies the Gaifman condition.

Proof.

$$
P_\mu(\exists x \varphi x) = \mu(\{M \in \mathcal{M} : M \models \exists x \varphi x\})
$$

$$
= \mu(\{M \in \mathcal{M} : M \models \varphi(c), \text{ for some } c \in C\})
$$

$$
= \sup \left\{ \mu(M \in \mathcal{M} : M \models \bigvee_{i \leq n} \varphi(c_i) \mid c_1, \ldots, c_n \in C \right\}
$$

$$
= \sup \left\{ P_\mu(\bigvee_{i \leq n} \varphi(c_i)) \mid c_1, \ldots, c_n \in C \right\}
$$

Note that one may need to use countable additivity in this argument!
First-Order Probability Logic

- Add probability operators to the first-order language:
  \[ \pi(\varphi) \]

  and corresponding atomic formulas, for \( a, b, c, \ldots \in \mathbb{R} \), e.g.:
  \[ a\pi(\varphi)^2 + b\pi(\psi) = c \]
We can also express, e.g., certain facts about conditional probability:

\[ \pi(\varphi \mid \psi) \geq \frac{2}{3} \equiv 3\pi(\varphi \land \psi) \geq 2\pi(\psi) \]

Models of this language are tuples:

\[ \langle \mathcal{M}, \mathcal{E}, \mu, \mathcal{M} \rangle \]

- \( \mathcal{M} \) is a set of models with \( \mathcal{E} \) a \( \sigma \)-algebra over \( \mathcal{M} \);
- \( (\mathcal{M}, \mathcal{E}, \mu) \) is a probability space;
- \( \mathcal{M} \in \mathcal{M} \) is a distinguished model in \( \mathcal{M} \).

But note that in the background there are also real numbers!
Note on Terminology

Beware, we now have three different types of probabilities. We have mostly been using the following terminology:

\[ \mu(E) \] — measure of event \( E \), as part of a probability space

\[ P(\varphi) \] — meta-language symbol for probability of a sentence \( \varphi \)

\[ \pi(\varphi) \] — object language symbol for probability of a sentence \( \varphi \)

Tomorrow we will add a sentence forming modal operator \( P^>_r \) in the object language, where \( P^>_r \varphi \) is not a term but a sentence.
Language and Interpretation

- Two sorts of variables: object variables \( x, y, z, \ldots \), and field variables \( x, y, z, \ldots \). Plus field constants \( 0 \) and \( 1 \) and plus and times.

- Field terms:

\[
\begin{align*}
t & ::= x \mid y \mid \cdots \mid 0 \mid 1 \mid \pi(\varphi) \mid t \ast t \mid t + t
\end{align*}
\]

- Then the language is given as follows:

\[
\varphi ::= P(x_1, \ldots, x_n) \mid x = y \mid t_1 = t_2 \mid \varphi \land \varphi \mid \neg \varphi \mid \forall x \varphi \mid \forall x \varphi
\]

- Interpretation of field terms:

\[
\left[ \pi(\varphi) \right]_{\langle \mathfrak{m}, \mathcal{E}, \mu, \mathcal{M} \rangle} = \mu(\{ \mathcal{M}' \in \mathfrak{M} : \mathcal{M}' \models \varphi \})
\]
This is clearly a very expressive language. For instance, all of the probability axioms come out as validities in the object language.

If $\Gamma \models \varphi$ in this language, that means any probability distribution that satisfies all the probabilistic (and other) requirements in $\Gamma$, also satisfies whatever requirement $\varphi$ specifies.

One might also like to study maximum entropy inference in this setting. See, e.g., Plaskin (2002).
In general, the problem of determining validity in this language is undecidable. Indeed, the theory of models in this language is not even axiomatizable. (See Abadi & Halpern 1989.)

However, if we assume the domain is finite, it is possible to give an axiomatization, and prove decidability.

One can also obtain a complete axiomatization by allowing non-standard reals (see Bacchus 1988).
Axiomatization

- The axioms and rules of classical first-order logic.

- The axioms for **real closed fields**, e.g., that multiplication and addition are commutative and associated, that every odd degree polynomial has a root, etc. (Tarski)

- $\varphi \rightarrow (\pi(\varphi) = 1)$, provided the only predicate symbols occurring in $\varphi$ occur only as subformulas of some $\pi(\psi)$.

- $\pi(\varphi) \geq 0$

- $\pi(\varphi) = \pi(\varphi \land \neg \psi) + \pi(\varphi \land \psi)$

- From $\varphi \leftrightarrow \psi$, infer $\pi(\varphi) = \pi(\psi)$

- $\exists x_1 \ldots x_n \forall y \left(y = x_1 \lor \ldots \lor y = x_n\right)$
Theorem (Halpern 1990)

This logic is sound and complete with respect to probability models of domain size at most $n$. 
Another way of curtailing complexity:

- Still add probability operators to the first-order language:
  \[ \pi(\varphi) \]
  but corresponding atomic formulas, for \( a_1, \ldots, a_n, b \in \mathbb{R} \):
  \[ a_1 \pi(\varphi_1) + \ldots + a_n \pi(\varphi_n) > b \]

- Abbreviations:
  \[
  \begin{align*}
    \pi(\varphi) - \pi(\psi) > b & \equiv \pi(\varphi) + (-1)\pi(\psi) > b \\
    \pi(\varphi) > \pi(\psi) & \equiv \pi(\varphi) - \pi(\psi) > 0 \\
    \pi(\varphi) \leq b & \equiv \neg(\pi(\varphi) > b) \\
    \pi(\varphi) \geq b & \equiv (-1)\pi(\varphi) \leq -b \\
    \pi(\varphi) = b & \equiv (\pi(\varphi) \geq b) \land (\pi(\varphi) \leq b)
  \end{align*}
  \]
Then the language is given as follows:

\[
\varphi ::= P(x_1, \ldots, x_n) \mid a_1 \pi(\varphi) + \ldots + a_n \pi(\varphi) > b \mid \\
a = b \mid x = y \mid \varphi \land \varphi \mid \neg \varphi \mid \forall x \varphi \mid \exists x \varphi
\]

The crucial truth clause:

\[
\langle M, E, \mu, M \rangle \models a_1 \pi(\varphi_1) + \ldots + a_n \pi(\varphi_n) > b \iff \\
\llbracket a_1 \rrbracket \ast \mu(\llbracket \varphi_1 \rrbracket) + \cdots + \llbracket a_1 \rrbracket \ast \mu(\llbracket \varphi_n \rrbracket) > b \text{ in } \mathbb{R}
\]

**Theorem (Halpern 2003)**

Adding axioms for linear inequalities gives a sound and complete axiomatization for this language.
Measures on the Domain

- This language seems ill-suited for reasoning about what might be called a **chance setup**, e.g., the probability that a randomly chosen student in Tübingen being German is high.

- E.g., we should not expect this formula to have high probability:

\[
\forall x (\text{StudentTbg}(x) \rightarrow \text{German}(x))
\]

- To deal with these kinds of statements, Bacchus (1988) proposed a slightly different language, with operators

\[
\pi_x(\varphi(x)) > b
\]

meaning, roughly, the probability of a randomly chosen \( x \) satisfying \( \varphi \) is greater than \( b \).
Otherwise the language is just the same as before.

Models, however, are simpler:

\[ \mathcal{M} = \langle D, I, \nu_D, \gamma \rangle \]

with \( \nu : D \rightarrow [0, 1] \) a map such that \( \sum_{d \in D} \nu(d) = 1 \).

The interpretation of a probability subformula \( \pi_x(\varphi) \) is given as:

\[ \llbracket \pi_x(\varphi) \rrbracket_{\mathcal{M}} = \nu(\{ d \in D : \mathcal{M}[d/x] \models \varphi \}) \]

The rest of interpretation is as before.
Complexity

- This logic, too, is **undecidable and unaxiomatizable** (Abadi & Halpern 1989).

- However, once again, by restricting to models of bounded size, the class becomes axiomatizable.

- Alternatively, we can restrict the language to linear inequalities and include a truth clause:

  \[ M \models a_1 \pi_x(\varphi_1) + \ldots + a_n \pi_x(\varphi_n) > b \text{ iff } \]
  \[ [a_1] \star [\pi_x(\varphi_1)] + \ldots + [a_n] \star [\pi_x(\varphi_n)] > b \text{ in } \mathbb{R} \]

  This system is again axiomatizable.
Theorem (Halpern 1990)

This class of models of size at most $n$ is axiomatized by the principles and rules of first order logic, those of real closed fields, and:

- $\forall x \varphi \rightarrow (\pi_x(\varphi) = 1)$
- $\pi_x(\varphi) \geq 0$
- $\pi_x(\varphi) = \pi_x(\varphi \land \psi) + \pi_x(\varphi \land \neg \psi)$
- $\pi_x(\varphi) = \pi_y(\varphi[y/x])$
- From $\varphi \leftrightarrow \psi$, infer $\pi_x(\varphi) = \pi_x(\psi)$
- $\exists x_1 \ldots x_n \forall y (y = x_1 \lor \ldots \lor y = x_n)$. 
Of course, these two languages (and classes of models) can also be combined, so that one can reason, e.g., about such statements:

\[ \pi \left( \pi_x \left( \text{German}(x) \mid \text{StudentTbg}(x) \right) \geq \frac{1}{2} \right) \geq \frac{3}{4} \]

Combining the two axiomatizations gives completeness for the combined language, again for models of bounded size.

See Halpern (1990) for details.
Summary and Preview

- Untyped lambda calculus with a coin flip operation provides a representation language for defining a very rich class of probabilistic processes and models, even if it does not straightforwardly give rise to a logical system as we would typically think of it.

- There is a canonical way of extending a measure on the quantifier free sentences of a first-order language to all sentences.

- An important theme is the difference between defining a probability $P$ on a logical language and defining a measure $\mu$ on a class of logical models. Today we focused on the latter, including rich first-order languages for talking about these measures.

- These logics are highly complex. Tomorrow and Friday we will look at much simpler languages for logical reasoning about probability.