```
begin
    import Pkg
    # activate the shared project environment
    Pkg.activate(Base.current_project())
    using Omega, Distributions, UnicodePlots, OmegaExamples
end
```

```
  Activating project at `~/Documents/GitHub/Omega.jl/OmegaExamples`
```

```
using GraphPlot, Graphs, Colors
```

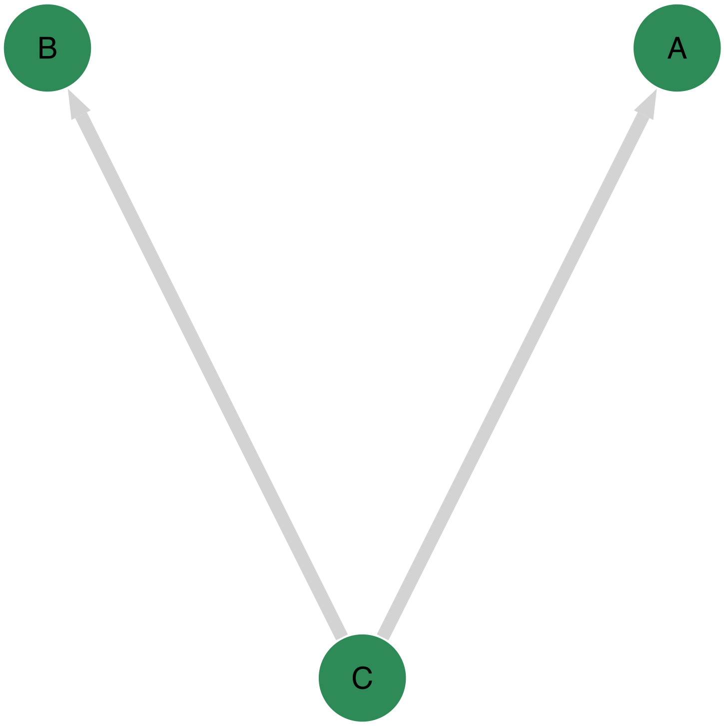
From *A Priori* Dependence to Conditional Dependence

The relationships between causal structure and statistical dependence become particularly interesting and subtle when we look at the effects of additional observations or assumptions. Events that are statistically dependent a priori may become independent when we condition on some observation; this is called *screening off*. Also, events that are statistically independent a priori may become dependent when we condition on observations; this is known as *explaining away*. The dynamics of screening off and explaining away are extremely important for understanding patterns of inference—reasoning and learning—in probabilistic models.

Screening off

Screening off refers to a pattern of statistical inference that is quite common in both scientific and intuitive reasoning. If the statistical dependence between two events `A` and `B` is only indirect, mediated strictly by one or more other events `C`, then conditioning on (observing) `C` should render `A` and `B` statistically independent. This can occur if `A` and `B` are connected by one or more causal chains, and all such chains run through the set of events `C`, or if `C` comprises all of the common causes of `A` and `B`.

For instance, let's look again at our common cause example, this time assuming that we *already* know the value of `C`:




```
C = 6020731536906764827@Distributions.Bernoulli{Float64}(p=0.5)
```

```julia
C = @~ Bernoulli()
```

```
B =
ifelseₚ(6020731536906764827@Distributions.Bernoulli{Float64}(p=0.5), 5449093775101357536@Di
```

```julia
B = ifelse.(C, (@~ Bernoulli(0.5)), (@~ Bernoulli(0.9)))
```

```
A =
ifelseₚ(6020731536906764827@Distributions.Bernoulli{Float64}(p=0.5), 8498248644661488725@Di
```

```julia
A = ifelse.(C, (@~ Bernoulli(0.1)), (@~ Bernoulli(0.4)))
```

```
B_cond_A (generic function with 1 method)
```

```julia
B_cond_A(A_val) = B |ᶜ (C .& (A .== A_val))
```

Histogram of `B` conditioned on when `A` is `true`:

```
        ┌                                        ┐
  false ┤■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 494
   true ┤■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 506
        └                                        ┘
```

```julia
viz(randsample(B_cond_A(true), 1000))
```

Histogram of `B` conditioned on when `A` is `false`:

```
        ┌                                        ┐
  false ┤■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 486
   true ┤■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ 514
        └                                        ┘
```

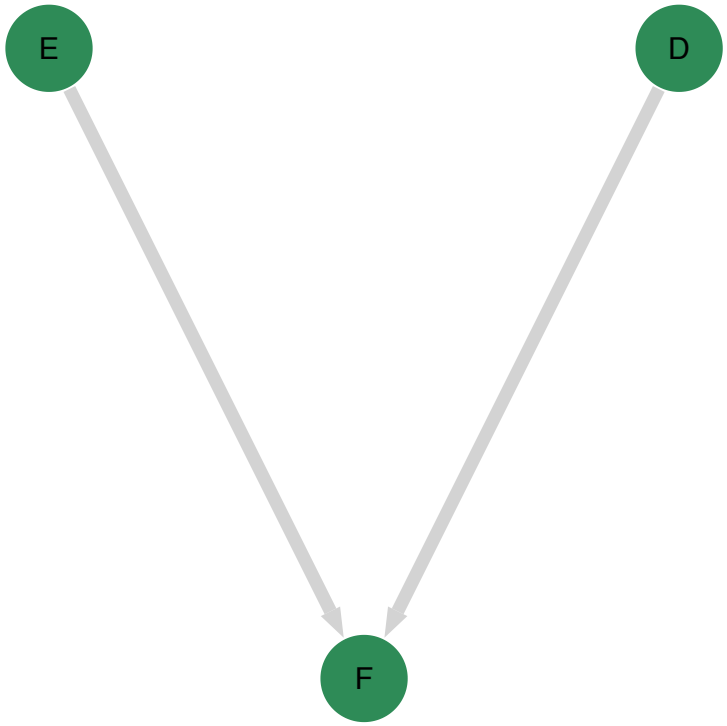
```julia
viz(randsample(B_cond_A(false), 1000))
```

We see that `A` and `B` are statistically independent given knowledge of `C`.

Screening off is a purely statistical phenomenon. For example, consider the the causal chain model, where `A` directly causes `C`, which in turn directly causes `B`. Here, when we condition on `C` – the event that mediates an indirect causal relation between `A` and `B` – `A` and `B` are still causally dependent in our model of the world: it is just our beliefs about the states of `A` and `B` that become uncorrelated. There is also an analogous causal phenomenon. If we can actually manipulate or intervene on the causal system, and set the value of `C` to some known value, then `A` and `B` become both statistically and causally independent (by intervening on `C`, we break the causal link between `A` and `C`).

Explaining away

"Explaining away" (Pearl, 1988) refers to a complementary pattern of statistical inference which is somewhat more subtle than screening off. If two events `D` and `E` are statistically (and hence causally) independent, but they are both causes of one or more other events `F`, then conditioning on (observing) `F` can render `D` and `E` statistically dependent. Here is an example where `D` and `E` have a common *effect*:




```
D = 5554423190373557753@Distributions.Bernoulli{Float64}(p=0.5)
```

```
1 D = @~ Bernoulli()
```

```
E = 6987100154786239906@Distributions.Bernoulli{Float64}(p=0.5)
```

```
1 E = @~ Bernoulli()
```

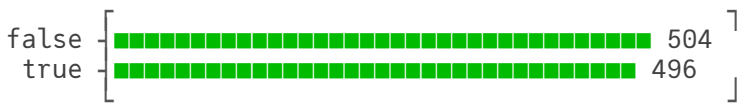
```
F = 
ifelse_p(|_p(5554423190373557753@Distributions.Bernoulli{Float64}(p=0.5), 6987100154786239906
```

```
1 F = ifelse.((D .| E), (@~ Bernoulli(0.9)), (@~ Bernoulli(0.2)))
```

```
E_cond_D (generic function with 1 method)
```

```
1 E_cond_D(D_val) = E |ᶜ (F .& (D .== D_val))
```

Histogram of `E` conditioned on `D` when `D` is `true`:




```
viz(randsample(E_cond_D(true), 1000))
```

Histogram of `E` conditioned on `D` when `D` is `false`:


```
viz(randsample(E_cond_D(false), 1000))
```

As with screening off, we only induce statistical dependence from learning about `F`, not causal dependence: when we observe `F`, `D` and `E` remain causally independent in our model of the world; it is our beliefs about `D` and `E` that becomes correlated.

The most typical pattern of explaining away we see in causal reasoning is a kind of *anti-correlation*: the probabilities of two possible causes for the same effect increase when the effect is observed, but they are conditionally anti-correlated, so that observing additional evidence in favor of one cause should lower our degree of belief in the other cause. (This pattern is where the term explaining away comes from.) However, the coupling induced by conditioning on common effects depends on the nature of the interaction between the causes, it is not always an anti-correlation. Explaining away takes the form of an anti-correlation when the causes interact in a roughly disjunctive or additive form: the effect tends to happen if any cause happens; or the effect happens if the sum of some continuous influences exceeds a threshold. The following simple mathematical examples show this and other patterns.

Suppose we condition on observing the sum of two integers drawn uniformly from 0 to 9:

```
int_1 = 8126598191684806461@Distributions.DiscreteUniform(a=0, b=9)
```

```
int_1 = @~ DiscreteUniform(0, 9)
```

```
int_2 = -2007371679756655155@Distributions.DiscreteUniform(a=0, b=9)
```

```
int_2 = @~ DiscreteUniform(0, 9)
```

```
ints = #7 (generic function with 1 method)
```

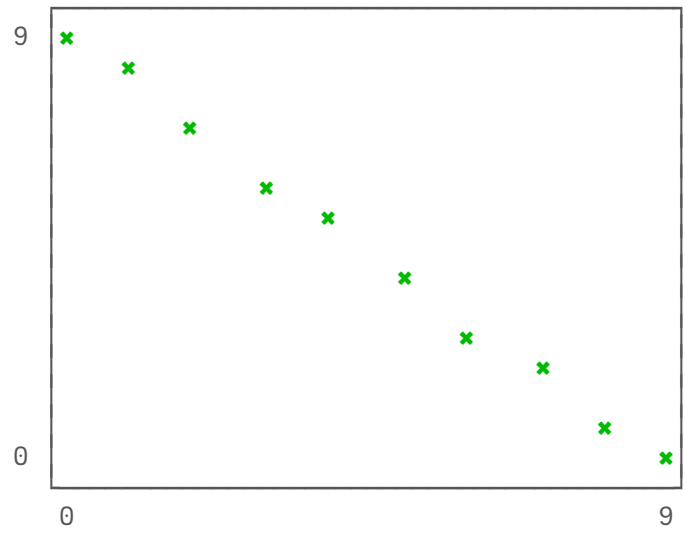
```
ints = @joint int_1 int_2
```

```
sum_cond =
  Conditional(#7 (generic function with 1 method), ==ₚ(+ₚ(8126598191684806461@Distributions.I
```

```
sum_cond = ints |ᶜ (int_1 .+ int_2 .== 9)
```

```
val =
  [(int_1 = 4, int_2 = 5), (int_1 = 5, int_2 = 4), (int_1 = 9, int_2 = 0), (int_1 = 9, int_2 = 0),
```

```
val = randsample(sum_cond, 1000)
```




```
1 scatterplot([v.int_1 for v in val], [v.int_2 for v in val], marker = :xcross)
```

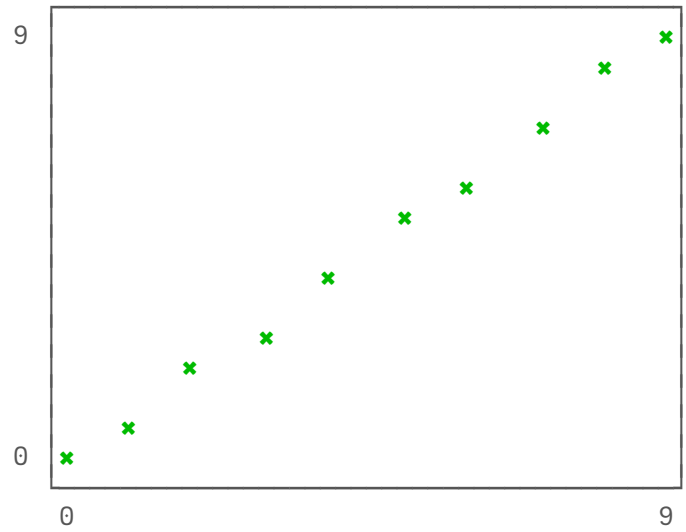
This gives perfect anti-correlation in conditional inferences for `int_1` and `int_2`. But suppose we instead condition on observing that `int_1` and `int_2` are equal:

```
eq_cond =
  Conditional(#7 (generic function with 1 method), ==ₚ(8126598191684806461@Distributions.Dis
```

```
1 eq_cond = ints |ᶜ (int_1 .== int_2)
```

```
val_eq =
  [(int_1 = 6, int_2 = 6), (int_1 = 6, int_2 = 6), (int_1 = 2, int_2 = 2), (int_1 = 8, int_2 = 8),
```

```
1 val_eq = randsample(eq_cond, 1000)
```




```
1 scatterplot([v.int_1 for v in val_eq], [v.int_2 for v in val_eq], marker = :xcross)
```

Now, of course, `int_1` and `int_2` go from being independent a priori to being perfectly correlated in the conditional distribution. Try out these other conditions to see other possible patterns of conditional dependence for a priori independent functions:

- `pw(abs, (int_1 -ₚ int_2)) <ₚ 2`
- `(int_1 +ₚ int_2 >=ₚ 9) &ₚ (int_1 +ₚ int_2 <=ₚ 11)`
- `pw(abs, (int_1 -ₚ int_2)) ==ₚ 2`

- `pw(%, (A -ₚ B), 10) ==ₚ 3` # $(int_1 - int_2) \% 10 == 3$
- `pw(%, int_1, 2) ==ₚ pw(%, int_2, 2)` # $int_1 \% 2 == int_2 \% 2$

Non-monotonic Reasoning

The medical scenario is a great model to explore screening off and explaining away. In this model `smokes` is statistically dependent on several symptoms—`cough`, `chest_pain`, and `shortness_of_breath`—due to a causal chain between them mediated by `lung_disease`. We can see this easily by conditioning on these symptoms and looking at `smokes`:

One reason explaining away is an important phenomenon in probabilistic inference is that it is an example of non-monotonic reasoning. In formal logic, a theory is said to be monotonic if adding an assumption (or formula) to the theory never reduces the set of conclusions that can be drawn. Most traditional logics (e.g. First Order) are monotonic, but human reasoning does not seem to be. For instance, if I tell you that Tweety is a bird, you conclude that he can fly; if I now tell you that Tweety is an ostrich you retract the conclusion that he can fly. Over the years many non-monotonic logics have been introduced to model aspects of human reasoning. One of the first reasons that probabilistic reasoning with Bayesian networks was recognized as important for AI was that it could perspicuously capture these patterns of reasoning (Pearl, 1988).

Another way to think about monotonicity is by considering the trajectory of our belief in a specific proposition, as we gain additional relevant information. In traditional logic, there are only three states of belief: true, false, and unknown (when neither a proposition nor its negation can be proven). As we learn more about the world, maintaining logical consistency requires that our belief in any proposition only move from unknown to true or false. That is our "confidence" in any conclusion only increases (and only does so in one giant leap from unknown to true or false).

In a probabilistic approach, by contrast, belief comes in a whole spectrum of degrees. We can think of confidence as a measure of how far our beliefs are from a uniform distribution—how close to the extremes of 0 or 1. In probabilistic inference, unlike in traditional logic, our confidence in a proposition can both increase and decrease. Even fairly simple probabilistic models can induce complex explaining-away dynamics that lead our degree of belief in a proposition to reverse directions multiple times as observations accumulate.

Example: Medical Diagnosis

```
smokes = 4297649084372236043@Distributions.Bernoulli{Float64}(p=0.2)
```

```
1 smokes = @~ Bernoulli(0.2)
```

```
lung_disease =
|ₚ(&ₚ(4297649084372236043@Distributions.Bernoulli{Float64}(p=0.2), -8426695402003269665@Dis
```

```
1 lung_disease = (smokes .& @~ Bernoulli(0.1)) .| @~ Bernoulli(0.001)
```

```
cold = -3659207040176488798@Distributions.Bernoulli{Float64}(p=0.02)
```

```
1 cold = @~ Bernoulli(0.02)
```

```
cough =
|ₚ(&ₚ(-3659207040176488798@Distributions.Bernoulli{Float64}(p=0.02), 17506384107399864320@Di
```

```
cough = pw(|,
    (cold .& @~ Bernoulli()),
    (lung_disease .& @~ Bernoulli()),
    @~ Bernoulli(0.001))
```

```
fever =
|ₚ(&ₚ(-3659207040176488798@Distributions.Bernoulli{Float64}(p=0.02), 30510889531898598320@Di
```

```
fever = (cold .& @~ Bernoulli(0.3)) .| @~ Bernoulli(0.01)
```

```
chest_pain =
|ₚ(&ₚ(|ₚ(&ₚ(4297649084372236043@Distributions.Bernoulli{Float64}(p=0.2), -84266954020032696
```

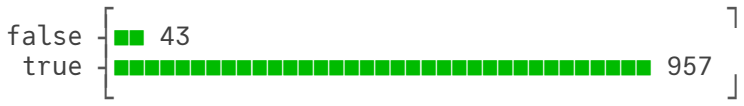
```
chest_pain = (lung_disease .& @~ Bernoulli(0.2)) .| @~ Bernoulli(0.01)
```

```
shortness_of_breath =
|ₚ(&ₚ(|ₚ(&ₚ(4297649084372236043@Distributions.Bernoulli{Float64}(p=0.2), -84266954020032696
```

```
shortness_of_breath = (lung_disease .& @~ Bernoulli(0.2)) .| @~ Bernoulli(0.01)
```

```
smokes_cond_c_cp_sob =
  Conditional(4297649084372236043@Distributions.Bernoulli{Float64}(p=0.2), &ₚ(|ₚ(&ₚ(-365920
```

```
smokes_cond_c_cp_sob = smokes |ᶜ pw(&, cough, chest_pain, shortness_of_breath)
```




```
viz(randsample(smokes_cond_c_cp_sob, 1000))
```

The conditional probability of smokes is much higher than the base rate, 0.2, because observing all these symptoms gives strong evidence for smoking. See how much evidence the different symptoms contribute by dropping them out of the conditioning set. (For instance, try conditioning on `cough &ₚ chest_pain`, or just `cough`; you should observe the probability of `smokes` decrease as fewer symptoms are observed.)

Now, suppose we condition also on knowledge about the function that mediates these causal links: `lung_disease`. Is there still an informational dependence between these various symptoms and `smokes`? In the Inference below, try adding and removing various symptoms (`cough`, `chest_pain`, `shortness_of_breath`) but maintaining the observation `lung_disease`:

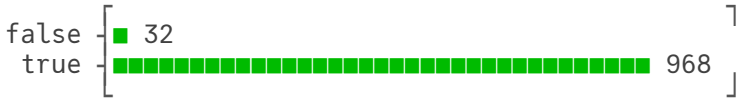
```
smokes_cond_c_cp_sob_ld =
  Conditional(4297649084372236043@Distributions.Bernoulli{Float64}(p=0.2), &ₚ(|ₚ(&ₚ(429764908
```

```
smokes_cond_c_cp_sob_ld =
    smokes |ᶜ pw(&, lung_disease, cough, chest_pain, shortness_of_breath)
```

You should see an effect of whether the patient has lung disease on conditional inferences about smoking—a person is judged to be substantially more likely to be a smoker if they have lung disease than otherwise—but there are no separate effects of chest pain, shortness of breath, or cough over and above the evidence provided by knowing whether the patient has lung-disease. The intermediate variable lung disease screens off the root cause (smoking) from the more distant effects (coughing, chest pain and shortness of breath).

Here is a concrete example of explaining away in our medical scenario. Having a cold and having lung disease are a priori independent both causally and statistically. But because they are both causes of coughing if we observe `cough` then `cold` and `lung_disease` become statistically dependent. That is, learning something about whether a patient has `cold` or `lung_disease` will, in the presence of their common effect `cough`, convey information about the other condition. `cold` and `lung_disease` are a priori independent, but conditionally dependent given `cough`.

To illustrate, observe how the probabilities of `cold` and `lung_disease` change when we observe `cough` is `true`:




```
viz(randsample(smokes_cond_c_cp_sob_ld, 1000))
```

```
ld_and_cold = #17 (generic function with 1 method)
```

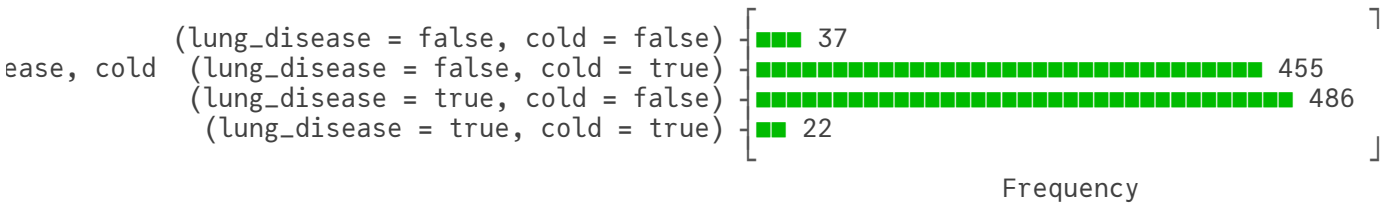
```
ld_and_cold = @joint lung_disease cold
```

```
ld_and_cold_cond =
  Conditional(#17 (generic function with 1 method), |ₚ(&ₚ(-3659207040176488798@Distributions
```

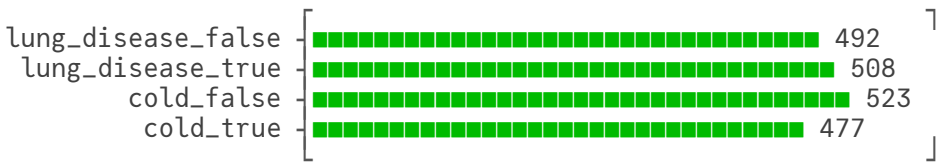
```
ld_and_cold_cond = ld_and_cold |ᶜ cough
```

```
cond_cough_samples =
  [(lung_disease = false, cold = true), (lung_disease = true, cold = false), (lung_disease = true
```

```
cond_cough_samples = randsample(ld_and_cold_cond, 1000)
```




```
viz(cond_cough_samples)
```




```
viz_marginals(cond_cough_samples)
```

Both cold and lung disease are now far more likely than their baseline probability: the probability of having a cold increases from 2% to around 50%; the probability of having lung disease also increases from 2.1% to around 50%.

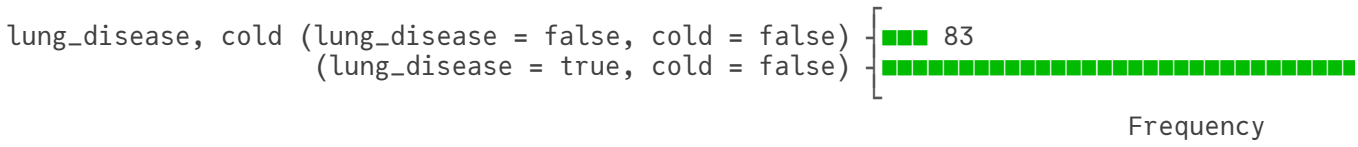
Now suppose we also learn that the patient does *not* have a cold.

```
cond_cough_not_cold =
  Conditional(#17 (generic function with 1 method), &p(|p(&p(-3659207040176488798@Distributi
```

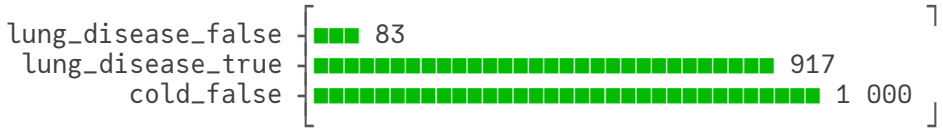
```
1 cond_cough_not_cold = ld_and_cold |ᶜ (cough .& .!(cold))
```

```
cond_cough_not_cold_samples =
  [(lung_disease = true, cold = false), (lung_disease = true, cold = false), (lung_disease = true
```

```
1 cond_cough_not_cold_samples = randsample(cond_cough_not_cold, 1000)
```




```
1 viz(cond_cough_not_cold_samples)
```




```
1 viz_marginals(cond_cough_not_cold_samples)
```

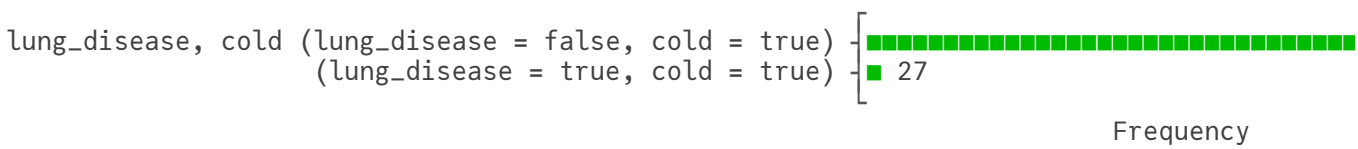
The probability of having lung disease increases dramatically. If instead we had observed that the patient does have a cold, the probability of lung cancer returns to its base rate of 2.1%

```
cond_cough_and_cold =
  Conditional(#17 (generic function with 1 method), &p(|p(&p(-3659207040176488798@Distributi
```

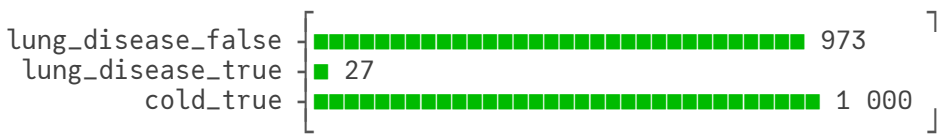
```
1 cond_cough_and_cold = ld_and_cold |ᶜ (cough .& cold)
```

```
cond_cough_cold_samples =
  [(lung_disease = false, cold = true), (lung_disease = false, cold = true), (lung_disease = fals
```

```
1 cond_cough_cold_samples = randsample(cond_cough_and_cold, 1000)
```




```
1 viz(cond_cough_cold_samples)
```




```
1 viz_marginals(cond_cough_cold_samples)
```

This is the conditional statistical dependence between lung disease and cold, given cough: Learning that the patient does in fact have a cold "explains away" the observed cough, so the alternative of lung disease decreases to a much lower value — roughly back to its 1 in a 1000 rate in the general population. If on the other hand, we had learned that the patient does not have a cold, so the most likely alternative to lung disease is not in fact available to "explain away" the observed cough, which raises the conditional probability of lung disease dramatically. As an exercise, check that if we remove

the observation of coughing, the observation of having a cold or not has no influence on our belief about lung disease; this effect is purely conditional on the observation of a common effect of these two causes.

Explaining away effects can be more indirect. Instead of observing the truth value of cold, a direct alternative cause of cough, we might simply observe another symptom that provides evidence for cold, such as fever. Compare these conditions using the above program to see an "explaining away" conditional dependence in belief between `fever` and `lung_disease`.

Example: Trait Attribution

A familiar example of rich patterns of inference comes from reasoning about the causes of students' success and failure in the classroom. Imagine yourself in the position of an interested outside observer —a parent, another teacher, a guidance counselor or college admissions officer—in thinking about these conditional inferences. If a student doesn't pass an exam, what can you say about why he failed? Maybe he doesn't do his homework, maybe the exam was unfair, or maybe he was just unlucky?

```
fair_exam = 7481845899745545208@Distributions.Bernoulli{Float64}(p=0.8)
```

```
fair_exam = @~ Bernoulli(0.8)
```

```
does_homework = -3134688556986477107@Distributions.Bernoulli{Float64}(p=0.8)
```

```
does_homework = @~ Bernoulli(0.8)
```

```
pass_prob (generic function with 1 method)
```

```
function pass_prob(ω)
    if fair_exam(ω)
        return does_homework(ω) ? 0.9 : 0.4
    else
        return does_homework(ω) ? 0.6 : 0.2
    end
end
```

```
pass = 6214270630802985957@#1
```

```
pass = @~ Bernoulli(Variable(pass_prob))
```

We write `Variable(pass_prob)` instead of just `pass_prob` to specify that `pass_prob` is of a `Variable` type.

```
fair_exam_cond_not_pass =
  Conditional(7481845899745545208@Distributions.Bernoulli{Float64}(p=0.8), !ₚ(62142706308029
```

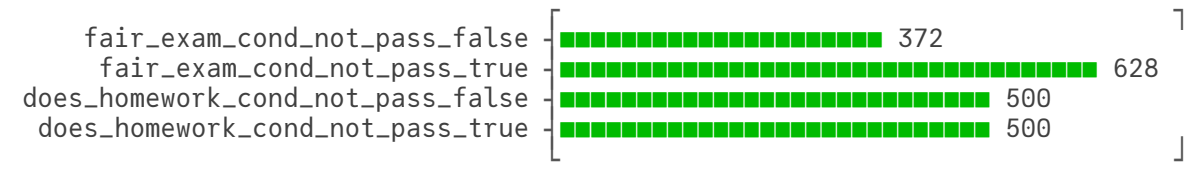
```
fair_exam_cond_not_pass = fair_exam |ᶜ .!(pass)
```

```
does_homework_cond_not_pass =
  Conditional(-3134688556986477107@Distributions.Bernoulli{Float64}(p=0.8), !ₚ(6214270630802
```

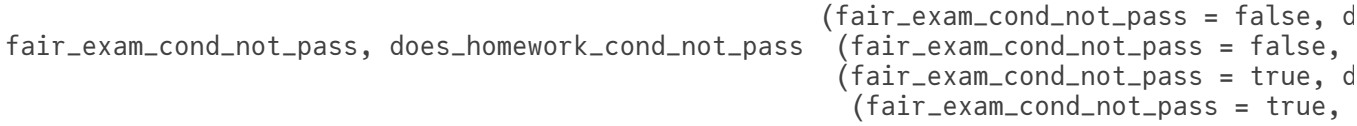
```
does_homework_cond_not_pass = does_homework |ᶜ .!(pass)
```

```
samples_cond_not_pass =
  [(fair_exam_cond_not_pass = true, does_homework_cond_not_pass = true), (fair_exam_cond_not_pass
```

```
samples_cond_not_pass =
    randsample((@joint fair_exam_cond_not_pass does_homework_cond_not_pass), 1000)
```




```
viz_marginals(samples_cond_not_pass)
```




```
viz(samples_cond_not_pass)
```

Now what if you have evidence from several students and several exams? We first re-write the above model to allow many students and exams:

```
fair_exam_m (generic function with 1 method)
```

```
fair_exam_m(exam) = exam ~ Bernoulli(0.8)
```

```
does_homework_m (generic function with 1 method)
```

```
does_homework_m(student) = student ~ Bernoulli(0.8)
```

```
pass_m (generic function with 1 method)
```

```
pass_m(exam, student) = @~ Bernoulli(
    ifelse.(fair_exam_m(exam),
        ifelse.(does_homework_m(student), 0.9, 0.4),
        ifelse.(does_homework_m(student), 0.6, 0.2)))
```

```
101
```

```
begin
    bill = 1
    exam1 = 101
end
```

```
does_homework_bill (generic function with 1 method)
```

```
does_homework_bill(condition) = does_homework_m(bill) |ᶜ condition
```

```
fair_exam_exam1 (generic function with 1 method)
```

```
fair_exam_exam1(condition) = fair_exam_m(exam1) |ᶜ condition
```

```
joint (generic function with 1 method)
```

```
joint(condition) = ω -> (fair_exam_exam1 = fair_exam_exam1(condition)(ω),
    does_homework_bill = does_homework_bill(condition)(ω))
```

```
p (generic function with 1 method)
```

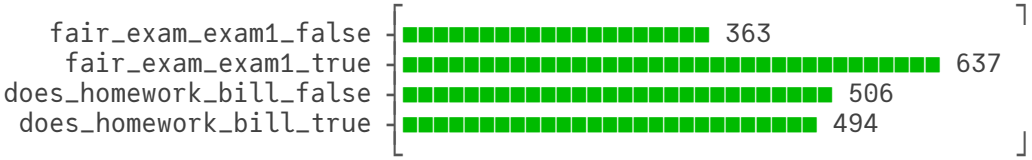
```
p(condition) =
    randsample(joint(condition), 1000)
```

```
c = !ₚ(350560614854146639 5@#1,)
```

```
c = .!(pass_m(exam1, bill))
```

```
                                   (fair_exam_exam1 = false, does_homework_bill = false
fair_exam_exam1, does_homework_bill  (fair_exam_exam1 = false, does_homework_bill = true
                                     (fair_exam_exam1 = true, does_homework_bill = false
                                      (fair_exam_exam1 = true, does_homework_bill = true
```

```
viz(p(c))
```




```
viz_marginals(p(c))
```

Initially we observe that Bill failed exam 1. A priori, we assume that most students do their homework and most exams are fair, but given this one observation it becomes somewhat likely that either the student didn't study or the exam was unfair.

Notice that we have set the probabilities in the pass function to be asymmetric: whether a student does homework has a greater influence on passing the test than whether the exam is fair. This in turns means that when inferring the cause of a failed exam, the model tends to attribute it to the person property (not doing homework) over the situation property (exam being unfair). This asymmetry is an example of the *fundamental attribution bias* (Ross, 1977): we tend to attribute outcomes to personal traits rather than situations. However there are many interacting tendencies (for instance the direction of this bias switches for members of some east-asian cultures). How could you extend the model to account for these interactions?

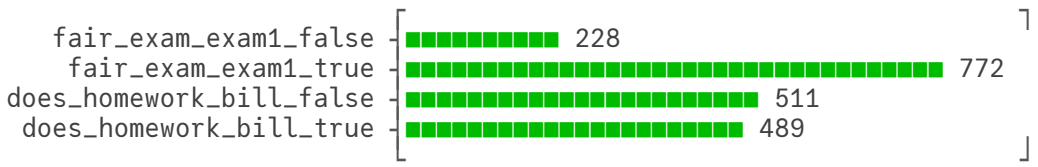
See how conditional inferences about Bill and exam 1 change as you add in more data about this student or this exam, or additional students and exams.

```
c1 = &ₚ(!ₚ(350560614854146639 5@#1,), !ₚ(350560614854146639 5@#1,))
```

```
c1 = .!(pass_m(exam1, bill)) .& .!(pass_m(bill, 102))
```

```
                                   (fair_exam_exam1 = false, does_homework_bill = false
fair_exam_exam1, does_homework_bill  (fair_exam_exam1 = false, does_homework_bill = true
                                     (fair_exam_exam1 = true, does_homework_bill = false
                                      (fair_exam_exam1 = true, does_homework_bill = true
```

```
viz(p(c1))
```




```
viz_marginals(p(c1))
```

Try using each of the below expressions as the condition (`c1`) for the above inference. Try to explain the different inferences that result at each stage. What does each new piece of the larger data set contribute to your intuition about Bill and exam 1?

- `pw(&, !ₚ(pass_m(bill, exam1)), !ₚ(pass_m(2, exam1)), !ₚ(pass_m(3, exam1)))`
- `pw(&, !ₚ(pass_m(bill, exam1)), !ₚ(pass_m(bill, 102)), !ₚ(pass_m(2, exam1)), !ₚ(pass_m(3, exam1)))`
- `pw(&, !ₚ(pass_m(bill, exam1)), !ₚ(pass_m(2, exam1)), (pass_m(2, 102)), (pass_m(2, 103)), (pass_m(2, 104)), (pass_m(2, 105)), !ₚ(pass_m(3, exam1)), (pass_m(3, 102)), (pass_m(3, 103)), (pass_m(3, 104)), (pass_m(3, 105)))`
- `pw(&, !ₚ(pass_m(bill, exam1)), (pass_m(2, exam1)), (pass_m(3, exam1)))`
- `pw(&, !ₚ(pass_m(bill, exam1)), (pass_m(2, exam1)), (pass_m(2, 102)), (pass_m(2, 103)), (pass_m(2, 104)), (pass_m(2, 105)), (pass_m(3, exam1)), (pass_m(3, 102)), (pass_m(3, 103)), (pass_m(3, 104)), (pass_m(3, 105)))`
- `pw(&, !ₚ(pass_m(bill, exam1)), !ₚ(pass_m(bill, 102)), (pass_m(2, exam1)), (pass_m(2, 102)), (pass_m(2, 103)), (pass_m(2, 104)), (pass_m(2, 105)), (pass_m(3, exam1)), (pass_m(3, 102)), (pass_m(3, 103)), (pass_m(3, 104)), (pass_m(3, 105)))`
- `pw(&, !ₚ(pass_m(bill, exam1)), !ₚ(pass_m(bill, 102)), (pass_m(bill, 103)), (pass_m(bill, 104)), (pass_m(bill, 105)), (pass_m(2, exam1)), (pass_m(2, 102)), (pass_m(2, 103)), (pass_m(2, 104)), (pass_m(2, 105)), (pass_m(3, exam1)), (pass_m(3, 102)), (pass_m(3, 103)), (pass_m(3, 104)), (pass_m(3, 105)))`

This example is inspired by the work of Harold Kelley (and many others) on causal attribution in social settings (Kelley, 1973). Kelley identified three important dimensions of variation in the evidence, which affect the attributions people make of the cause of an outcome. These three dimensions are: Persons—is the outcome consistent across different people in the situation?; Entities—is the outcome consistent for different entities in the situation?; Time—is the outcome consistent over different episodes? These dimensions map onto the different sets of evidence we have just seen.

As in this example, people often have to make inferences about entities and their interactions. Such problems tend to have dense relations between the entities, leading to very challenging explaining away problems. These inferences often come very naturally to people, yet they are computationally difficult. Perhaps these are important problems that our brains have specialized somewhat to solve, or perhaps that they have evolved general solutions to these tough inferences.

Example: Of Blickets and Blocking

A number of researchers have explored children's causal learning abilities by using the "blicket detector" (Gopnik and Sobel, 2000): a toy box that will light up when certain blocks, the blickets, are put on top of it. Children are shown a set of evidence and then asked which blocks are blickets. For instance, if block A makes the detector go off, it is probably a blicket. Ambiguous patterns are particularly interesting. Imagine that blocks A and B are put on the detector together, making the detector go off; it is fairly likely that A is a blicket. Now B is put on the detector alone, making the detector go off; it is now less plausible that A is a blicket. This is called "backward blocking", and it is an example of explaining away.

We can capture this set up with a model in which each block has a persistent "blicket-ness" property, and the causal power of the block to make the machine go off depends on its blicketness. Finally, the machine goes off if any of the blocks on it is a blicket (but noisily).

```
blicket (generic function with 1 method)
```

```
blicket(block) = block ~ Bernoulli(0.4)
```

```
power (generic function with 1 method)
```

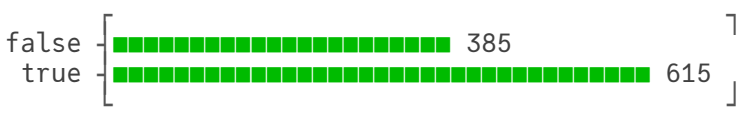
```
power(block) = ifelse.(blicket(block), 0.9, 0.05)
```

```
machine (generic function with 1 method)
```

```
function machine(blocks)
    if isempty(blocks)
        return @~ Bernoulli(0.05)
    else
        return (@~ Bernoulli(power(blocks[1]))) .| machine(blocks[2:end])
    end
end
```

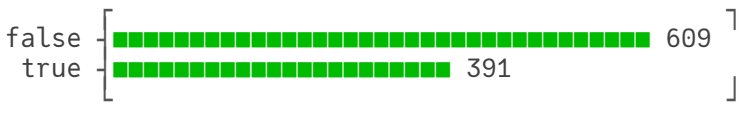
```
blicket_cond =
  Conditional(1@Distributions.Bernoulli{Float64}(p=0.4), |ₚ(-92127953260751650 44@#1, |ₚ(-921
```

```
blicket_cond = blicket(1) |ᶜ machine([1, 2])
```




```
viz(randsample(blicket_cond, 1000))
```

The backward blocking scenario described above:




```
viz(randsample(blicket(1) |ᶜ machine([2]), 1000))
```

Sobel et al. (2004) tried this with children, finding that four year-olds perform similarly to the model: evidence that B is a blicket explains away the evidence that A and B made the detector go away.