Inference, Models and Simulation for Complex Systems CSCI 7000-003
Lecture 4 9 September 2010
Prof. Aaron Clauset

1 Some general comments

There are several distinct but rich literatures on the general topics of model plausibility and model comparison. We won't cover them all. For instance, Bayesian approaches for model comparison are widely used in machine learning and computational biology, while classic frequentist methods and techniques based on out-of-sample prediction (cross-validation) grew out of physics and statistics. More recently, computer science has introduced approaches based on compression and minimum description lengths (MDL). Bayesian, frequentist and MDL approaches are based on parametric models and use the likelihood function to make comparisons. A few approaches eschew the likelihood function, which can be useful if a model is too complicated to be translated into a likelihood function or if it's unreasonable to make strong iid assumptions about your empirical data. One of these alternatives uses goodness-of-fit statistics to make comparisons; another uses model-free (non-parametric) operations like shuffling or sorting the data.

All of these approaches wrestle with questions of statistical power (the probability of reaching a false conclusion), model simplicity (Occam's razor), and generalizability (out-of-sample prediction). Many of their differences in how they do so are methodological but the fundamental points of disagreement are actually philosophical, especially between the different likelihood-based methods. That is, the different approaches implicitly embody different preferences about what makes a *good* model. Unfortunately, there has been too little work on trying to understand or resolve these differences, and too much effort spent bashing largely legitimate alternative statistical approaches.

In some ways, the mathematical nature of many of these approaches has encouraged their misuse. That is, their quantitative nature has allowed some people to believe that the hard problems of epistemology in science (How can we be confident that we've learned something true about the world from our data?) have been solved and their particular question of interest can be answered correctly by simply applying some kind of scientific-method algorithm. These tools, however, all come with assumptions about data that are invariably wrong to some degree. Real data are rarely iid and when the tool's assumptions aren't met, it can yield spurious or misleading results. The best methods fail gracefully and conservatively when their assumptions aren't met, that is, they err on the side of saying "I don't know". For complex systems, statistical methods are indispensable, since the data are often noisy, biased, dirty or too large to look at any other way, but they can only support, not replace, careful thinking, careful tests and better experiments.

2 Model plausibility, p-values and statistical power

A useful tool for testing model *plausibility* is to ask whether the observed deviations of the data from the model can be explained as mere statistical fluctuations. This can be done by using a classic statistical *hypothesis test* in a backwards way.

The forwards way to do this detects deviations so large that they are unlikely to have been generated by noise. This is the basis for tests of normality, which underlie many industrial and clinical statistical tests. These tests generally rely on a statistical measure of "closeness" $D = f(\{x_i\}, A)$ between the data $\{x_i\}$ and the model A. The Kolmogorov-Smirnov goodness-of-fit statistic is one such measure (see Lecture 3). First, we measure D for the empirical data (denoted D^*) and then ask what fraction of synthetic data sets generated by A would yield a *larger* deviation, i.e., we want to compute $\Pr(D \geq D^*)$. Figure 1a illustrates this idea.

Here's a pseudo-code algorithm for doing the test. Let x be the data set, n be the number of observations it contains, N be a very large number and A be our model.

```
{x_i}       = empirical data
theta*      = maximum likelihood parameter estimate of A fitted to {x_i}
compute D*  = f({x_i},theta*)
for i=1:N
    {y_i}        = n synthetic observations, drawn from A
    theta-s[i]   = maximum likelihood parameter estimate of A fitted to {y_i}
    compute D[i] = f({y_i},theta-s[i])
end
p = sum(D>D*)/N
```

The inner loop runs a Monte Carlo simulation of $\Pr(D)$; it does this by repeating many many times the same three steps we performed on our empirical data, (i) data acquisition (generation), (ii) model fitting or parameter estimation, and (iii) deviation measurement. If the model parameters are known *a priori*, e.g., the model has no parameters or their values are known independently of our data, then we can skip the estimation step.[1]

If our data $\{x_i\}$ are truly drawn uniformly from the generating process A, then the distribution of p-values $\Pr(p)$ will be uniform on the unit interval $(0, 1)$. The hypothesis that A generated $\{x_i\}$ is rejected if $p < \alpha$, where α is an arbitrary value. Conventionally, $\alpha = 0.1$ or some other small value. This means that about 10% of the time, we falsely reject A as the generating process.

[1] If we use θ^* in our calculation of the deviation of the data from the model, then we bias the deviations high and thus make D^* look smaller than it should be.

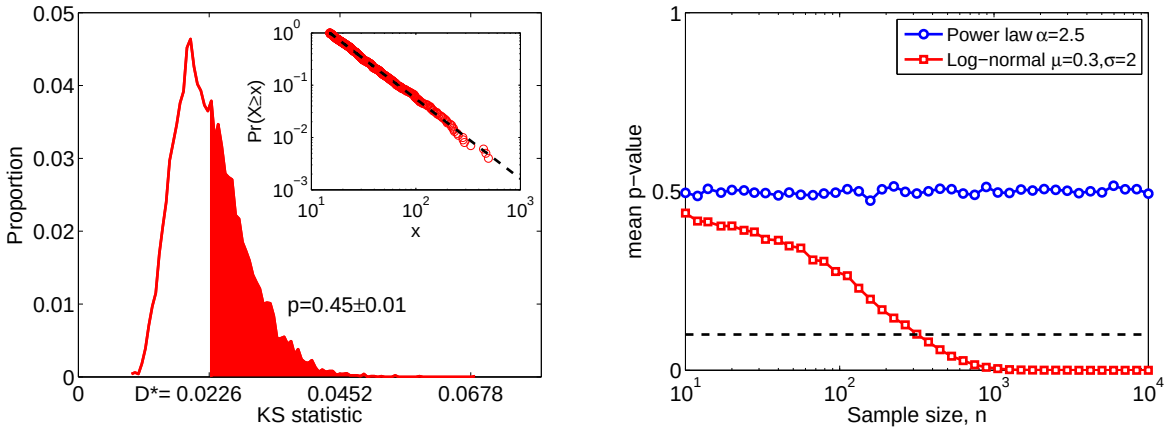

Figure 1: (a) The distribution of deviations $\Pr(D)$ for $n = 1000$ observations drawn from a power-law distribution with $x_{\min} = 15$ and $\alpha = 2.5$; the inset shows a single synthetic data set with its maximum likelihood model ($\hat{\alpha} = 2.530 \pm 0.048$), and D^* marks the corresponding deviation. (b) The behavior of the average p-value for data drawn from a power-law distribution and from a log-normal distribution, when both are fitted with a power-law form.

To demonstrate this more concretely, let's again use the power-law distribution (denoted model A) along with the log-normal distribution (denoted model B), which is defined as

$$\Pr(x) \propto \frac{1}{x} \mathrm{e}^{-\frac{(\ln x - \mu)^2}{\sqrt{2\sigma^2}}} \quad , \tag{1}$$

where μ and σ are parameters. For large σ, log-normals have very heavy tails, and thus have broad regions that are approximately power-law-like. This effect is more pronounced if n is relatively small. We can see this if we take the logarithm of both sides of Eq. (1):

$$\ln \Pr(x) \propto -\ln x - \frac{1}{\sqrt{2\sigma^2}} (\ln x - \mu)^2 \quad ,$$

where the quadratic term effectively controls the amount of curvature observed in the ccdf. When σ is large, this term is small and $\ln \Pr(x) \approx -\ln x$, which is a power-law form.

Suppose we generate data using B but we mistakenly fit them using A. In this case, we would expect $p \to 0$ as $n \to \infty$. However, suppose we generated data using A and we fit them using A. Now, p will be distributed uniformly and we'll get non-trivial p-values, i.e., $p > 0.1$, 90% of the time. Figure 1b shows precisely this experiment, which confirms that it performs as desired.[2]

[2]See Section 4 below for Matlab code that produces Fig. 1b.

For small values of n, however, data from B actually yields non-trivial p-values meaning that they look plausibly power-law distributed. This illustrates the point of *statistical power*, which is the ability of a test to distinguish distinct generating processes. In this case, as $n \to \infty$ the statistical power increases, i.e., the test better distinguishes power-law and log-normal data, but for small sample sizes, the statistical power is low and we cannot distinguish A from B. (Note: there's also some dependency on the particular choices of θ_A and θ_B.)

A few more cautionary comments: In this backwards version of the hypothesis test, getting a non-trivial p-value does not mean that the power-law model is correct and that our data do in fact follow a power-law distribution. Rather, it simply means that we could not falsify that notion. To really prove the hypothesis requires coming up with a mechanistic model that produces power-law distributed data, showing that the models assumptions hold up empirically, and that it produces data that are statistically indistinguishable from the empirical data. Once we have a candidate mechanistic model M, regardless of how complicated it is, we can replace A with M in the procedure above and compute its p-value. As the model being tested becomes more complicated, some aspects of this procedure become trickier. For instance, (i) the parameter estimation can be more difficult, especially if the likelihood function becomes rugged, (ii) the model's computational complexity can become problematic, and (iii) it can become more ambiguous how define a useful "deviation" between model and data as the model makes more and different kinds of predictions.

In the forward version, which is far more common than the backwards version, a small p-value, which would lead you to reject the candidate model A as the generating process, may not in fact be particularly helpful. That is, a small p-value is only useful if the model A is scientifically meaningful. We can always rig the test in favor of rejection (in favor of finding "statistical significance") by choosing a ridiculously unrealistic candidate model that is sure to fail for any reasonably complex process. Tiny p-values like $p < 10^{-5}$, which are often reported in the biomedical literature, should actually be a warning sign that the candidate model was completely unrealistic to begin with because it only generates observations like the empirical data once in 100,000 or more tials.

3 Model comparison

Model comparison, a.k.a. model selection, is typically done by comparing the likelihood of the data $\{x_i\}$ under model A to its likelihood under model B (or, more generally, comparing it under a number of models $\{A_j\}$). Under a likelihood framework, the interpretation is clear: if the data are more likely under A, then A is a better explanation of the data; and conversely for B. The differences between the several likelihood-based approaches for model comparison come down to questions of (i) how to account for different numbers of parameters, (ii) how to account for increasing sample size, and (iii) how to account for fluctuations in the data themselves.

3.1 The Bayesian approach

The Bayesian approach to model comparison is increasingly popular, and can be very powerful. It can be used both to compare completely orthogonal models or to choose the number of parameters or model "complexity" within a general family of models.

The primary Bayesian assumption is that the data $\{x_i\}$ are fixed.[3] The fundamental quantity in Bayesian model comparison is the *marginal likelihood* (sometimes also called the "evidence"), which is simply the likelihood of the data integrated over all parameter choices:

$$\Pr(\{x_i\} \,|\, A) = \int_\theta \Pr(\{x_i\} \,|\, \theta) \Pr(\theta \,|\, A) \, \mathrm{d}\theta \ . \tag{2}$$

If we have two models A and B, then we can compare the marginal likelihoods of each, i.e., compare $\Pr(\{x_i\} \,|\, A)$ to $\Pr(\{x_i\} \,|\, B)$ and ask which is better (larger). This procedure generalizes to arbitrary numbers of models and to models with differing number of parameters. In fact, the whole point of the marginalization procedure is to eliminate the effect that different numbers of parameters have—that is, models with more parameters are more "complex" and thus more flexible, but, as a result, they assign lower likelihoods to all the data sets they can generate. In contrast, simpler models assign higher likelihoods to a smaller range of data sets, and thus should win under this kind of model comparison. Thus, marginalization in the Bayesian framework is a kind of formalization of Occam's razor. [4] (This idea is illustrated by a cartoon in Figure 2.)

Further, it can be shown that this approach to model comparison is asymptotically consistent when the true generating process is in the set of models we fit to the data, that is, given that $\{x_i\} \sim M$ and $M \in \{A_j\}$, then as $n \to \infty$, $\sup_j \Pr(\{x_i\} \,|\, A_j) \to M$.

3.1.1 Information Criteria

There are a number of "information criteria" that can be used within a likelihood framework to facilitate model comparison. These take the form of a penalty term that is subtracted from the log-likelihood to "correct" for the advantage that flexible models have in fitting data:

$$\ln \mathcal{L}_\mathrm{c} = \ln \mathcal{L} - f(k, n) \ , \tag{3}$$

[3] This assumptions is different from that of the frequentist method for model plausibility we saw in Section 2, which assumes the data $\{x_i\}$ are random variables, i.e., if you looked again, you would get different values drawn from the same generative model.

[4] Albert Einstein in 1933: "It can scarcely be denied that the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience." This is often paraphrased as "Models should be made as simple as possible, but no simpler." This notion that accuracy should not be sacrificed for simplicity is slightly different from that of Occam's razor, and so is sometimes called Einstein's razor to distinguish the two.

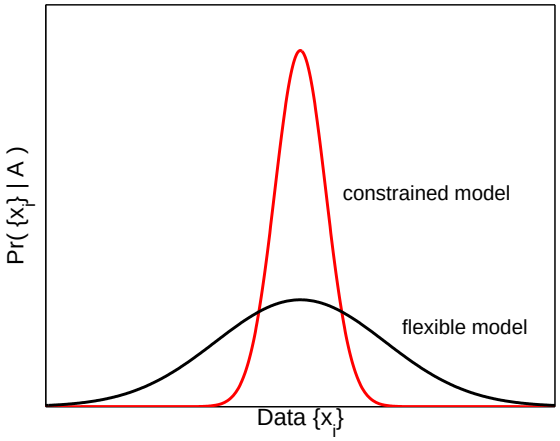

Figure 2: A cartoon of the difference between constrained models (low "complexity") and flexible models (high "complexity") in the range of data they can fit well (produce with high likelihood).

where $f(k, n)$ is a function that depends on the number of parameters in the model k, and possibly also on the number of observations n. Thus, the more complex or flexible a model is, the larger the penalty. For instance, the Bayesian Information Criterion (BIC) is defined as

$$f_{\mathrm{BIC}} = k \ln n \ . \tag{4}$$

There are several ways to mathematically derive the BIC, e.g., by considering an asymptotic Gaussian approximation to the shape of the likelihood function at the MLE's location. Intuitively, the functional form represents a kind of volume measure in the k-dimensional likelihood space, i.e., $\mathrm{e}^{f_{\mathrm{BIC}}} = n^k$, so that models with a more concentrated likelihood function get less of a penalty than "bigger" models (see again Fig. 2).

There are other information criteria, including "An Information Criterion" (AIC) due to Aikake (and thus, usually called Aikake's Information Criterion), which is defined as

$$f_{\mathrm{AIC}} = k \ , \tag{5}$$

and which also has a mathematical basis. The tiny difference in the functional form between the AIC and BIC points to the fact that there are differing philosophical preferences of what makes a good model underlying the mathematical analysis. For largely historical reasons, AIC is typically used in the ecological sciences, while BIC is used in other parts of biology, and marginalization is favored in machine learning.

3.1.2 Warning label (a.k.a. the model-comparison fallacy)

This framework is quite powerful but there's an important assumption that is not typically discussed in the literature. That is, this (and most other model comparison approaches) can only identify the true generating process M if that model is within the family being compared. If M is not in the set, marginalization will still pick some model, i.e., $\sup_j \Pr(\{x_i\} \mid A_j)$ is still perfectly well defined, but it will not warn us that the model it chooses is wrong. For instance, consider the following thought experiment:

Suppose Nature, in her infinite wisdom, uses model C to generate some empirical data $\{x_i\}$, which she then gives to us. But, being distracted by other issues, she neglects to tell us anything about model C. Now, attempting to recover this information, we think very hard and deeply and come up with two good-sounding candidate generating processes A and B. And, just to be completely clear, $C \notin \{A, B\}$. We then proceed through our model comparison algorithm and determine the likelihood of the data under A and the likelihood of the data under B. The result is that model A wins—the data were more likely to be generated by it than by the alternative. We then write up our results and share with the world.

What's wrong with this picture? The problem is that because model C was not in the set of models being considered, we arrived at a conclusion that doesn't tell us much about the true generating process. That is, our conclusion is technically correct but scientifically unhelpful. Even if model A is closer to the data than model B, it could still be very far away when compared with the true generating process C. Model comparison only produces *relative* measures of goodness and is unable to tell us when all models under consideration are terrible. By construction, it must choose a winner.

This behavior is not problematic if we're not interested in the true generating process. For instance, this is often the case in industrial settings where we're more concerned with squeezing our error rates as much as possible or where the true generating process is probably too complex to ever codify. However, in scientific settings, this is not the case and interpreting the results of a model comparison exercise as if they were the results of a model plausibility test, i.e., concluding that model A is the correct generating process, can lead to false conclusions.

3.2 The minimum description length principle

We won't cover the minimum-description length (MDL) version of model comparison. MDL is an alternative philosophy for choosing among models that is based on notions of model-based compressibility and information theory and it's also framed in terms of penalized likelihood functions. That is, there's some $f_{\mathrm{MDL}}(k, n)$ that you would attach to your log-likelihoods. If you're interested, see P.D. Grünwald, "the Minimum Description Length principle." MIT Press (2009).

3.3 Likelihood-ratio tests

This approach to model comparison is covered in the first problem set. The main difference between it and the other approaches is that, as a frequentist method, it assumes that the data $\{x_i\}$ are random variables and thus, small differences in likelihoods under two candidate models could be generated by statistical fluctuations. The upside of this assumption is that a likelihood ratio test (LRT) can say "I don't know" when asked which model is best; this is particularly useful if you want to avoid false conclusions. It, like all model-comparison techniques, cannot circumvent the fallacy of model comparison (see Section 3.1.2).

3.4 Cross-validation

Cross-validation can also be used to compare models and does so without using likelihood functions. This can be particularly useful when a model is too complicated to express as a likelihood function.

Cross-validation (CV) works by simulating an out-of-sample prediction experiment in which we "train" the models on some observed data and then score how well each does at predicting future data. We simulate this by taking our empirical data $\{x_i\}$ and splitting it into two sets $y = \{x_1, x_2, \ldots, x_k\}$ and $z = \{x_{k+1}, x_{k+2}, \ldots, x_n\}$. We then fit any necessary model parameters using only y, and then score each model on the likelihood of generating z. The key step in this process is to choose the partitioning of data between y and z randomly, and to repeat the process many times, averaging the results across the many trials. Here's a psuedo-code version of this algorithm. Let $f = |z|/|x|$ the fraction of data we try to predict out-of-sample and N be a large number:

```
for i=1:N
    for j=1:m                                        % for each of the m models
        {z_i}    = f*n observations from {x_i} chosen uniformly
        {y_i}    = the remaining observations from {x_i}
        fit model A_j to data {y_i}
        L[j,i]   = likelihood of {z_i} under A_j    % or other ‘‘score’’ function
    end
end
mean-score = mean(L,2)
winner     = sup(mean-score)    % inf if the smaller values are ‘‘better’’ scores
```

The leave-one-out (LOO) version of CV, where $f = 1/n$, can be shown mathematically to be choose the right "best" model under a wide variety of conditions.

3.5 Goodness-of-fit

We won't cover them in detail here, but another set of likelihood-free approaches for model comparison use notions of "closeness" between the model output and the data. If the measure of closeness

has certain mathematical properties, then choosing the model that maximizes closeness, e.g., minimizing the KS goodness-of-fit statistic, can be asymptotically accurate. For more information, see D. Pollard, "Empirical Processes: Theory and Applications." Institute of Mathematical Sciences (1990).

4 Matlab code

Here's Matlab code that produces Fig. 1b. The `xxxxxx` bits correspond to the random deviate generator for the power-law distribution that you derive as part of the first problem set; if you replace it with the correct expression, the code should run just fine. It will take a while (about 3 hours on my laptop) to finish as-is, but after each value of n, it will draw the current results in a figure so you can see how things evolve.

```
[xmin alpha] = deal(15,2.5);
[mu   sigma] = deal(0.3,2);

reps   = 1000;  % number of trials over which we average the p-value
N      = 100;   % number samples in Monte Carlo simulation of Pr(D)
nr     = unique(round(logspace(1,4,41)));                % sample sizes n

pvalx  = zeros(length(nr),reps);
pvaly  = zeros(length(nr),reps);

tic;
for i=1:length(nr)
    n = nr(i);                                           % set sample size
    for j=1:reps

        % (1) generate synthetic data for (i) PL (via transformation
        %     method) and (ii) LN (via rejection sampling)

        x = xxxxxxxxxxxxxxxxxxxx;                        % PL
        y = exp(mu+sigma.*randn(n,1)); y(y<xmin) = [];   % LN
        while length(y)<n                                %
            yy = exp(mu+sig.*randn(n,1));                % cheat
            y = [y; yy(yy>=xmin)];                       % drop values <xmin
            if length(y)>n, y = y(1:n); end;             % truncate down to n obs
        end;
```

```
    % (2) fit the PL model to each synthetic data set

    thetx = 1+n./sum(log(x./xmin));                  % PL
    thety = 1+n./sum(log(y./xmin));                  % LN

    % (3) calculate KS statistics for the fitted models

    cn    = (0:n-1)'./n;                             % EDF
    cx    = 1-(xmin./sort(x)).^(thetx-1);            % PL
    gofx  = max( abs(cn - cx) );                     %
    cy    = 1-(xmin./sort(y)).^(thety-1);            % LN
    gofy  = max( abs(cn - cy) );                     %

    % (4) Monte Carlo simulation

    Ds = zeros(N,3);                                 % stores simulated Pr(D)
    for k=1:size(Ds,1)
        qx      = xxxxxxxxxxxxxxxxxxxxx;             % PL
        thexx   = 1+n./sum(log(qx./xmin));           %
        cx      = 1-(xmin./sort(qx)).^(thexx-1);     %
        Ds(k,1) = max( abs(cn - cx) );               %

        qy      = xxxxxxxxxxxxxxxxxxxxx;             % LN
        theyy   = 1+n./sum(log(qy./xmin));           %
        cy      = 1-(xmin./sort(qy)).^(theyy-1);     %
        Ds(k,2) = max( abs(cn - cy) );               %
    end;
    pvalx(i,j) = sum(Ds(:,1)>=gofx)./size(Ds,1);     % compute and store PL p value
    pvaly(i,j) = sum(Ds(:,2)>=gofy)./size(Ds,1);     % compute and store LN p value

end;

fprintf('n[%i]\t[%4.2fm]\n',i,toc/60);               % write out our progress
figure(1); clf;                                      % draw progress in a figure
semilogx(nr(1:i),mean(pvalx(1:i,:),2),'bo-','LineWidth',2,'MarkerFaceColor',[1 1 1]);
hold on;
semilogx(nr(1:i),mean(pvaly(1:i,:),2),'rs-','LineWidth',2,'MarkerFaceColor',[1 1 1]);
semilogx([nr(1) nr(end)],[0.1 0.1],'k--','LineWidth',2); hold off;
set(gca,'YLim',[0 1],'YTick',[0 0.5 1],'FontSize',16);
xlabel('Sample size, n','FontSize',16); ylabel('mean p-value','FontSize',16);
```

```
    h=legend('Power law \alpha=2.5','Log-normal \mu=0.3,\sigma=2',1);
    set(h,'FontSize',14);
    drawnow;

end;
```