The Learning Problem and Regularization

Tomaso Poggio

9.520 Class 02

February 2011

About this class

Theme We introduce the learning problem as the problem of function approximation from sparse data. We define the key ideas of loss functions, empirical error and generalization error. We then introduce the Empirical Risk Minimization approach and the two key requirements on algorithms using it: generalization and stability. We then describe a key algorithm – Tikhonov regularization – that satisfies these requirements.

Math Required Familiarity with basic ideas in probability theory.

About this class

Theme We introduce the learning problem as the problem of function approximation from sparse data. We define the key ideas of loss functions, empirical error and generalization error. We then introduce the Empirical Risk Minimization approach and the two key requirements on algorithms using it: generalization and stability. We then describe a key algorithm – Tikhonov regularization – that satisfies these requirements.

Math Required Familiarity with basic ideas in probability theory.

Plan

- Setting up the learning problem: definitions
- Generalization and Stability
- Empirical Risk Minimization
- Regularization
- Appendix: Sample and Approximation Error

Data Generated By A Probability Distribution

We assume that there are an "input" space X and an "output" space Y. We are given a **training set** S consisting n samples drawn i.i.d. from the probability distribution $\mu(z)$ on $Z = X \times Y$:

$$(x_1, y_1), \ldots, (x_n, y_n)$$

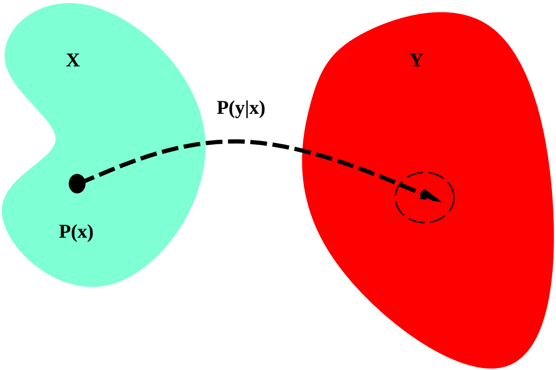
that is $z_1, \ldots, z_n$

We will use the **conditional probability of y given x**, written $p(y|x)$:

$$\mu(z) = p(x, y) = p(y|x) \cdot p(x)$$

It is crucial to note that we view $p(x, y)$ as **fixed** but **unknown.**

Probabilistic setting



Hypothesis Space

The **hypothesis space** $\mathcal{H}$ is the space of functions that we allow our algorithm to provide. For many algorithms (such as optimization algorithms) it is the space the algorithm is allowed to search. As we will see in future classes, it is often important to choose the hypothesis space as a function of the amount of data n available.

Learning As Function Approximation From Samples: Regression and Classification

The basic goal of **supervised learning** is to use the training set S to "learn" a function f_S that looks at a new x value x_{new} and predicts the associated value of y:

$$y_{pred} = f_S(x_{new})$$

If y is a real-valued random variable, we have **regression.**
If y takes values from an unordered finite set, we have **pattern classification.** In two-class pattern classification problems, we assign one class a y value of 1, and the other class a y value of -1.

Loss Functions

In order to measure goodness of our function, we need a **loss function** V. In general, we let $V(f, z) = V(f(x), y)$ denote the price we pay when we see x and guess that the associated y value is $f(x)$ when it is actually y.

Common Loss Functions For Regression

For regression, the most common loss function is square loss or L2 loss:

$$V(f(x), y) = (f(x) - y)^2$$

We could also use the absolute value, or L1 loss:

$$V(f(x), y) = |f(x) - y|$$

Vapnik's more general ϵ-insensitive loss function is:

$$V(f(x), y) = (|f(x) - y| - \epsilon)_+$$

Common Loss Functions For Classification

For binary classification, the most intuitive loss is the 0-1 loss:

$$V(f(x), y) = \Theta(-yf(x))$$

where $\Theta(-yf(x))$ is the step function and y is binary, eg $y = +1$ or $y = -1$. For tractability and other reasons, we often use the hinge loss (implicitely introduced by Vapnik) in binary classification:

$$V(f(x), y) = (1 - y \cdot f(x))_+$$

The learning problem: summary so far

There is an unknown **probability distribution** on the product space $Z = X \times Y$, written $\mu(z) = \mu(x, y)$. We assume that X is a compact domain in Euclidean space and Y a bounded subset of $\mathbb{R}$. The **training set** $S = \{(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)\} = \{z_1, ...z_n\}$

consists of n samples drawn i.i.d. from μ.

$\mathcal{H}$ is the **hypothesis space**, a space of functions $f : X \to Y$.

A **learning algorithm** is a map $L : Z^n \to \mathcal{H}$ that looks at S and selects from $\mathcal{H}$ a function $f_S : \mathbf{x} \to y$ such that $f_S(\mathbf{x}) \approx y$ *in a predictive way*.

Expected error, empirical error

Given a function f, a loss function V, and a probability distribution μ over Z, the **expected or true error** of f is:

$$I[f] = \mathbb{E}_z V[f, z] = \int_Z V(f, z) d\mu(z)$$

which is the **expected loss** on a new example drawn at random from μ.
We would like to make $I[f]$ small, but in general we do not know μ.
Given a function f, a loss function V, and a training set S consisting of n data points, the **empirical error** of f is:

$$I_S[f] = \frac{1}{n} \sum V(f, z_i)$$

Plan

- Setting up the learning problem: definitions
- **Generalization and Stability**
- Empirical Risk Minimization
- Regularization
- Appendix: Sample and Approximation Error

A reminder: convergence in probability

Let $\{X_n\}$ be a sequence of bounded random variables. We say that

$$\lim_{n\to\infty} X_n = X \textit{ in probability}$$

if

$$\forall\varepsilon > 0 \lim_{n\to\infty} \mathbb{P}\{|X_n - X| \geq \varepsilon\} = 0.$$

Generalization

A natural requirement for f_S is distribution independent **generalization**

$$\lim_{n\to\infty} |I_S[f_S] - I[f_S]| = 0 \textit{ in probability}$$

This is equivalent to saying that for each n there exists a ε_n and a $\delta(\varepsilon)$ such that

$$\mathbb{P}\{|I_{S_n}[f_{S_n}] - I[f_{S_n}]| \geq \varepsilon_n\} \leq \delta(\varepsilon_n),$$

with ε_n and δ going to zero for $n \to \infty$.
In other words, the training error for the solution must converge to the expected error and thus be a "proxy" for it. Otherwise the solution would not be "predictive".
A desirable additional requirement is **consistency**

$$\varepsilon > 0 \ \lim_{n\to\infty} \mathbb{P}\left\{ I[f_S] - \inf_{f\in\mathcal{H}} I[f] \geq \varepsilon \right\} = 0.$$

Finite Samples and Convergence Rates

More satisfactory results give guarantees for **finite number of points**: this is related to **convergence rates**.

Suppose we can prove that with probability at least $1 - e^{-\tau^2}$ we have

$$|I_S[f_S] - I[f_S]| \leq \frac{C}{\sqrt{n}}\tau$$

for some (problem dependent) constant C.

- The above result gives a convergence rate.
- If we fix ϵ, τ and solve for n the eq. $\epsilon = \frac{C}{\sqrt{n}}\tau$ we obtain the sample complexity:

$$n(\epsilon, \tau) = \frac{C^2\tau^2}{\epsilon^2}$$

the number of samples to obtain an error ϵ, with confidence $1 - e^{-\tau^2}$.

Remark: Finite Samples and Convergence Rates

Asymptotic results for generalization and consistency are valid *for any distribution* μ. It is impossible however to guarantee a given convergence rate independently of μ. This is Devroye's *No free lunch theorem*, see Devroye, Gyorfi, Lugosi, 1997, p112-113, Theorem 7.1). So there are rules that asymptotically provide optimal performance for any distribution. However, their finite sample performance is always extremely bad for some distributions.

So...how do we find good learning algorithms?

A learning algorithm should be well-posed, eg stable

In addition to the key property of generalization, a “good” learning algorithm should also be *stable*: f_S should depend continuously on the training set S. In particular, changing one of the training points should affect less and less the solution as n goes to infinity. Stability is a good requirement for the learning problem and, in fact, for any mathematical problem. We open here a small parenthesis on stability and well-posedness.

General definition of Well-Posed and Ill-Posed problems

A problem is **well-posed** if its solution:

- exists
- is unique
- depends continuously on the data (e.g. it is *stable*)

A problem is **ill-posed** if it is not well-posed. In the context of this class, well-posedness is mainly used to mean *stability* of the solution.

More on well-posed and ill-posed problems

Hadamard introduced the definition of ill-posedness. Ill-posed problems are typically inverse problems.
As an example, assume g is a function in Y and u is a function in X, with Y and X Hilbert spaces. Then given the linear, continuous operator L, consider the equation

$$g = Lu.$$

The direct problem is is to compute g given u; the inverse problem is to compute u given the data g. In the learning case L is somewhat similar to a "sampling" operation and the inverse problem becomes the problem of finding a function that takes the values

$$f(x_i) = y_i, i = 1, ...n$$

The inverse problem of finding u is well-posed when

- the solution exists,
- is unique and
- is *stable*, that is depends continuously on the initial data g.

Plan

- Setting up the learning problem: definitions
- Generalization and Stability
- **Empirical Risk Minimization**
- Regularization
- Appendix: Sample and Approximation Error

ERM

Given a training set S and a function space $\mathcal{H}$, empirical risk minimization (Vapnik introduced the term) is the class of algorithms that look at S and select f_S as

$$f_S = \arg\min_{f \in \mathcal{H}} I_S[f]$$

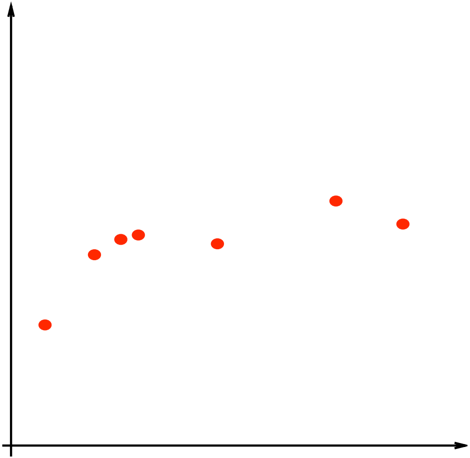
.
For example linear regression is ERM when $V(z) = (f(x) - y)^2$ and H is space of linear functions $f = ax$.

Generalization and Well-posedness of Empirical Risk Minimization

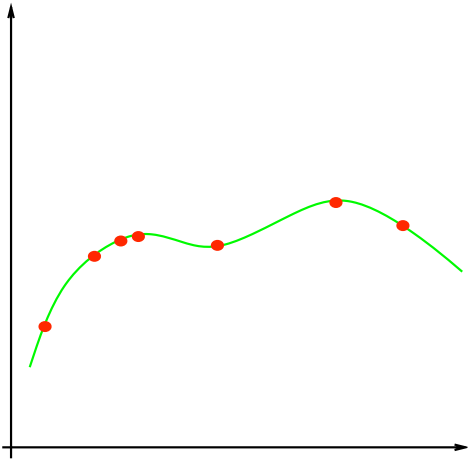
For ERM to represent a “good” class of learning algorithms, the solution should

- *generalize*
- exist, be unique and – especially – be *stable* (well-posedness).

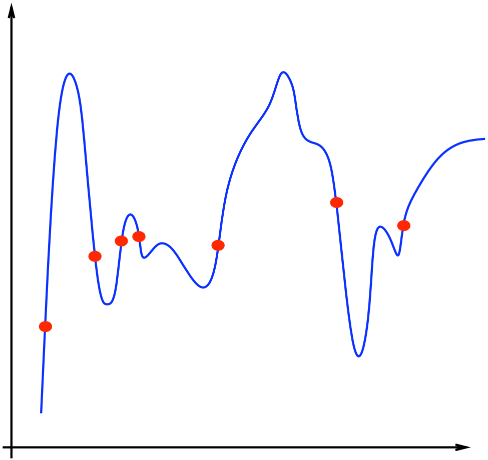
ERM and generalization: given a certain number of samples...



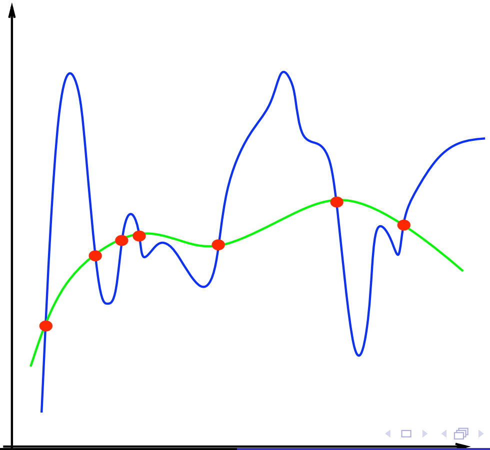
...suppose this is the “true” solution...



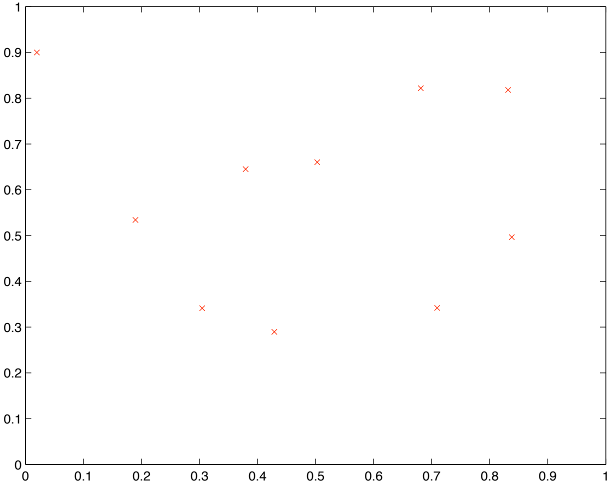
... but suppose ERM gives this solution.



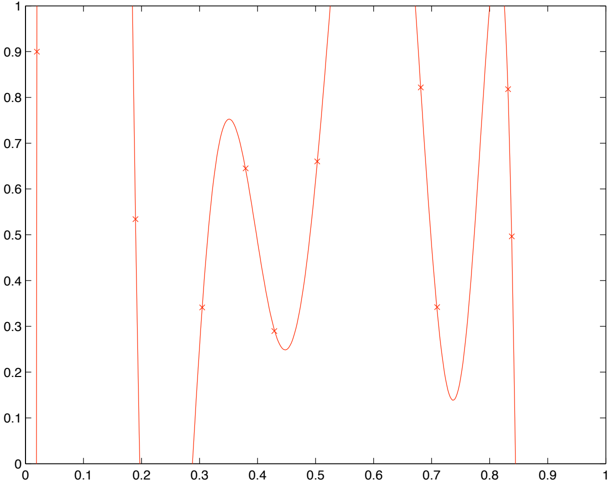
Under which conditions the ERM solution converges with increasing number of examples to the true solution? In other words...what are the conditions for generalization of ERM?



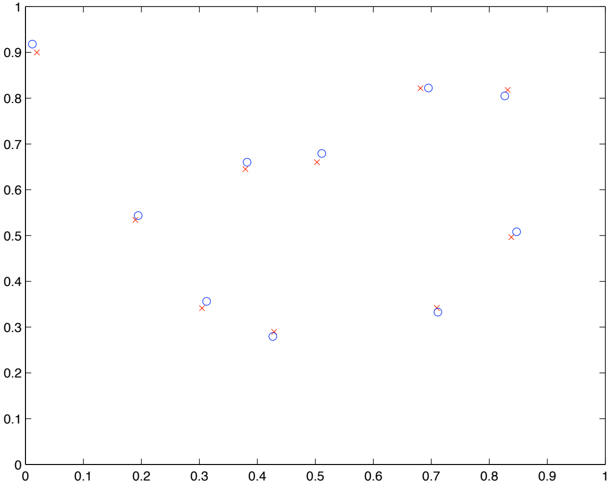
ERM and stability: given 10 samples...



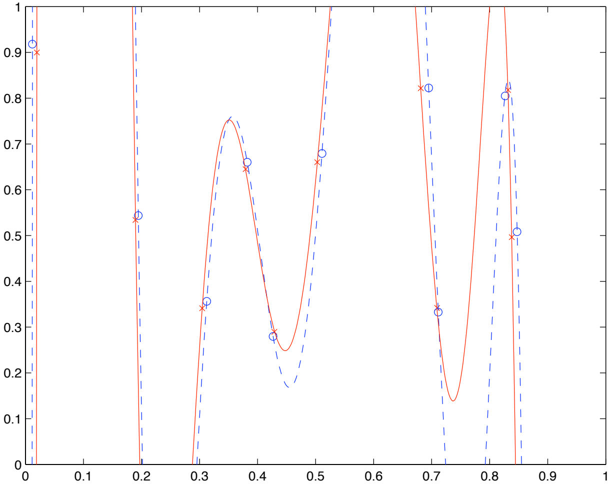
...we can find the smoothest interpolating polynomial (which degree?).



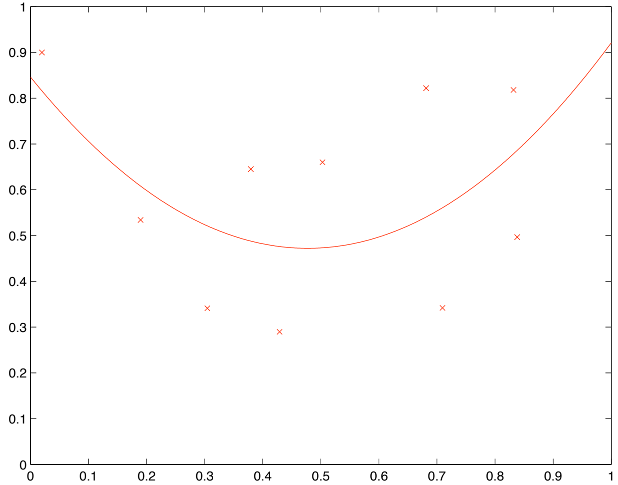
But if we perturb the points slightly...



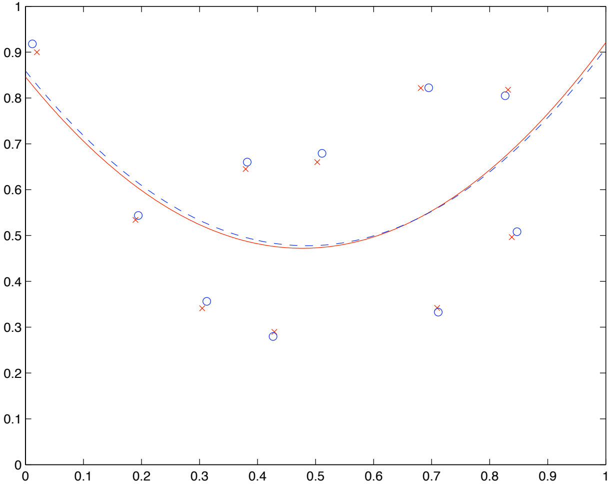
...the solution changes a lot!



If we restrict ourselves to degree two polynomials...



...the solution varies only a small amount under a small perturbation.



ERM: conditions for well-posedness (stability) and predictivity (generalization)

Since Tikhonov, it is well-known that a generally ill-posed problem such as ERM, can be guaranteed to be well-posed and therefore *stable* by an appropriate choice of $\mathcal{H}$. For example, compactness of $\mathcal{H}$ guarantees stability.
It seems intriguing that the *classical conditions for consistency of ERM* – thus quite a different property – consist of appropriately restricting $\mathcal{H}$. It seems that the same restrictions that make the approximation of the data stable, may provide solutions that generalize...

ERM: conditions for well-posedness (stability) and predictivity (generalization)

We would like to have a hypothesis space that yields generalization. Loosely speaking this would be a H for which the solution of ERM, say f_S is such that $|I_S[f_S] - I[f_S]|$ converges to zero in probability for n increasing.
Note that the above requirement is NOT the law of large numbers; the requirement for a fixed f that $|I_S[f] - I[f]|$ converges to zero in probability for n increasing IS the law of large numbers.

ERM: conditions for well-posedness (stability) and predictivity (generalization)

Theorem [Vapnik and Červonenkis (71), Alon et al (97), Dudley, Giné, and Zinn (91)]

A (necessary) and sufficient condition for generalization (and consistency) of ERM is that $\mathcal{H}$ is uGC.

Definition

$\mathcal{H}$ is a (weak) uniform Glivenko-Cantelli (uGC) class if

$$\forall \varepsilon > 0 \lim_{n \to \infty} \sup_{\mu} \mathbb{P}_S \left\{ \sup_{f \in \mathcal{H}} |I[f] - I_S[f]| > \varepsilon \right\} = 0.$$

ERM: conditions for well-posedness (stability) and predictivity (generalization)

- The theorem (Vapnik et al.) says that a proper choice of the hypothesis space $\mathcal{H}$ ensures generalization of ERM (and consistency since for ERM generalization is necessary and sufficient for consistency and viceversa). Other results characterize uGC classes in terms of measures of complexity or capacity of H (such as VC dimension).
- A separate theorem (Niyogi, Poggio et al., mentioned in the last class) guarantees also stability (defined in a specific way) of ERM. Thus with the appropriate definition of stability, *stability and generalization are equivalent for ERM.*

Thus the two desirable conditions for a learning algorithm – generalization and stability – are equivalent (and they correspond to the same constraints on $\mathcal{H}$).

Plan

- Setting up the learning problem: definitions
- Generalization and Stability
- Empirical Risk Minimization
- **Regularization**
- Appendix: Sample and Approximation Error

Regularization

Regularization (originally introduced by Tikhonov independently of the learning problem) ensures *well-posedness* and (because of the above argument) *generalization* of ERM by constraining the hypothesis space $\mathcal{H}$. The direct way – minimize the empirical error subject to f in a ball in an appropriate $\mathcal{H}$ – is called *Ivanov regularization*. The indirect way is *Tikhonov regularization* (which is not strictly ERM).

Ivanov and Tikhonov Regularization

ERM finds the function in $(\mathcal{H})$ which minimizes

$$\frac{1}{n}\sum_{i=1}^{n} V(f(x_i), y_i)$$

which in general – for arbitrary hypothesis space $\mathcal{H}$ – is *ill-posed*.

- Ivanov regularizes by finding the function that minimizes

  $$\frac{1}{n}\sum_{i=1}^{n} V(f(x_i), y_i)$$

  while satisfying $\mathcal{R}(f) \leq A$.
- Tikhonov regularization minimizes over the hypothesis space $\mathcal{H}$, for a fixed positive parameter γ, the regularized functional

  $$\frac{1}{n}\sum_{i=1}^{n} V(f(x_i), y_i) + \gamma\mathcal{R}(f). \quad (1)$$

$\mathcal{R}(f)$ is the regulirizer, a penalization on f. In this course we will mainly discuss the case $\mathcal{R}(f) = \|f\|_K^2$ where $\|f\|_K^2$ is the norm in the Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$, defined by the kernel

Tikhonov Regularization

As we will see in future classes

- Tikhonov regularization ensures well-posedness eg existence, uniqueness and especially *stability* (in a very strong form) of the solution
- Tikhonov regularization ensures generalization
- Tikhonov regularization is closely related to – but different from – Ivanov regularization, eg ERM on a hypothesis space $\mathcal{H}$ which is a ball in a RKHS.

Next Class

- In the next class we will introduce RKHS: they will be the hypothesis spaces we will work with.
- We will also derive the solution of Tikhonov regularization.

Plan

- Setting up the learning problem: definitions
- Generalization and Stability
- Empirical Risk Minimization
- Regularization
- Appendix: Sample and Approximation Error

Generalization, Sample Error and Approximation Error

Generalization error is $I_S[f_S] - I[f_S]$.
Sample error is $I[f_S] - I[f_{\mathcal{H}}]$
Approximation error is $I[f_{\mathcal{H}}] - I[f_0]$
Error is $I[f_S] - I[f_0] = (I[f_S] - I[f_{\mathcal{H}}]) + (I[f_{\mathcal{H}}] - I[f_0])$

Appendix: Target Space, Sample and Approximation Error

In addition to the hypothesis space $\mathcal{H}$, the space we allow our algorithms to search, we define...
The **target space** $\mathcal{T}$ is a space of functions, chosen a priori in any given problem, that is assumed to contain the "true" function f_0 that minimizes the risk. Often, $\mathcal{T}$ is chosen to be all functions in L_2, or all differentiable functions. Notice that the "true" function if it exists is defined by $\mu(z)$, which contains all the relevant information.

Sample Error (also called Estimation Error)

Let $f_{\mathcal{H}}$ be the function in $\mathcal{H}$ with the smallest true risk.
We have defined the **generalization error** to be $I_S[f_S] - I[f_S]$.
We define the **sample error** to be $I[f_S] - I[f_{\mathcal{H}}]$, the difference in true risk between the best function in $\mathcal{H}$ and the function in $\mathcal{H}$ we actually find. This is what we pay because our finite sample does not give us enough information to choose to the "best" function in $\mathcal{H}$. We'd like this to be small. *Consistency* – defined earlier – is equivalent to the sample error going to zero for $n \to \infty$.
A main goal in classical learning theory (Vapnik, Smale, ...) is "bounding" the generalization error. Another goal – for learning theory *and* statistics – is bounding the sample error, that is determining conditions under which we can state that $I[f_S] - I[f_{\mathcal{H}}]$ will be small (with high probability).
As a simple rule, we expect that if $\mathcal{H}$ is "well-behaved", then, as n gets large the sample error will become small.

Approximation Error

Let f_0 be the function in $\mathcal{T}$ with the smallest true risk.
We define the **approximation error** to be $I[f_{\mathcal{H}}] - I[f_0]$, the difference in true risk between the best function in $\mathcal{H}$ and the best function in $\mathcal{T}$. This is what we pay when $\mathcal{H}$ is smaller than $\mathcal{T}$. We'd like this error to be small too. In much of the following we can assume that $I[f_0] = 0$.
We will focus less on the approximation error in 9.520, but we will explore it.
As a simple rule, we expect that as $\mathcal{H}$ grows bigger, the approximation error gets smaller. If $\mathcal{T} \subseteq \mathcal{H}$ – which is a situation called *the realizable setting* –the approximation error is zero.

Error

We define the **error** to be $I[f_S] - I[f_0]$, the difference in true risk between the function we actually find and the best function in $\mathcal{T}$. We'd really like this to be small. As we mentioned, often we can assume that the **error** is simply $I[f_S]$.

The error is the sum of the sample error and the approximation error:

$$I[f_S] - I[f_0] = (I[f_S] - I[f_{\mathcal{H}}]) + (I[f_{\mathcal{H}}] - I[f_0])$$

If we can make both the approximation and the sample error small, the error will be small. There is a tradeoff between the approximation error and the sample error...

The Approximation/Sample Tradeoff

It should already be intuitively clear that making $\mathcal{H}$ big makes the approximation error small. This implies that we can (help) make the error small by making $\mathcal{H}$ big.
On the other hand, we will show that making $\mathcal{H}$ small will make the sample error small. In particular for ERM, if $\mathcal{H}$ is a uGC class, the generalization error and the sample error will go to zero as $n \to \infty$, but how quickly depends directly on the "size" of $\mathcal{H}$. This implies that we want to keep $\mathcal{H}$ as small as possible. (Furthermore, $\mathcal{T}$ itself may or may not be a uGC class.)
Ideally, we would like to find the optimal tradeoff between these conflicting requirements.

Generalization, Sample Error and Approximation Error

Generalization error is $I_S[f_S] - I[f_S]$.
Sample error is $I[f_S] - I[f_{\mathcal{H}}]$
Approximation error is $I[f_{\mathcal{H}}] - I[f_0]$
Error is $I[f_S] - I[f_0] = (I[f_S] - I[f_{\mathcal{H}}]) + (I[f_{\mathcal{H}}] - I[f_0])$