Practical ensemble modelling

François Bouttier, Météo-France, Nov 2021

This is a short course on practical ensemble prediction, its design and verification, plus some of its applications like probability forecast and sensitivity analysis.

Contents

Basic concepts

An ensemble is, by definition, a representation of uncertainty in our knowledge of a system. By "system" we mean here the state of a numerical model (e.g. the 3D atmosphere or ocean, or a smaller model down to a single environmental parameter). Because we only have incomplete and imperfect information about nature, there is always uncertainty about our representations of it. Examples:

- the analysis of the weather at any given time is imperfect because we cannot observe everything.
- the forecast of tomorrow's weather (rain patterns, fronts, storms, etc) is imperfect because it is made with incomplete forecast models initialized from imperfect analyses.
- using the ECMWF forecast system we can issue a temperature forecast for tonight at the top of the Everest (say, -43C), but there is some uncertainty around it (e.g. it

could be approximated by a gaussian distribution centered at -43C with a 4C standard deviation).

As shown by the above examples, **uncertainty is a property of a forecast or analysis system:** it will be different if another forecasting model is used. It depends on some prior knowledge (e.g. it changes as we process new observations). There is no such thing as "*the uncertainty of a weather event*", because the weather is what it is. The notion of uncertainty is relative to a modelling system. Mathematically, uncertainty is represented as a (potentially infinite) set of possible forecast values, each with its own probability.

Estimating the uncertainty on a parameter is fundamentally different from making the best possible forecast of it. Take for instance the temperature T at one point and time:

- the best forecast of T is unique: it is the actual real-world temperature at that point, which does not depend on the model or the time at which the forecast is made.
- the uncertainty on the forecast of T depends on the model used and time at which we forecast t. If we have a good forecasting system, it will set a high probability on the actual T.

Mathematically, uncertainty is modelled as a **distribution function**, which can be a gaussian, a range of possible values (i.e. a rectangular function), a finite set of possible values, or other functions.

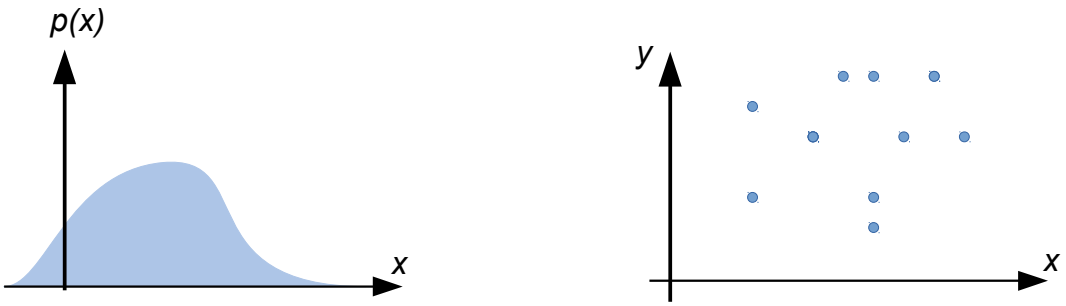


Figure: a 1-D bell-shaped continuous distribution, and a 2-D distribution of finite values

A numerical forecast is a process in which we take an estimate of the initial model state xa (typically, an analysis of the state of the atmosphere or ocean on a grid) and we feed it through a numerical simulation model M to generate a forecast $x_f = M(x_a)$ at some future time. What is the uncertainty of x_f ? Since it is a combination of uncertainties in xa and in the design of M (because models are not perfect), a theoretical way of computing the distribution of x_f is the following:

- estimate the distribution function of uncertainties on x_a
- apply all possible variants of M on a possible values of x_a
- the resulting set of x_f's has the distribution of the uncertainty on x_f

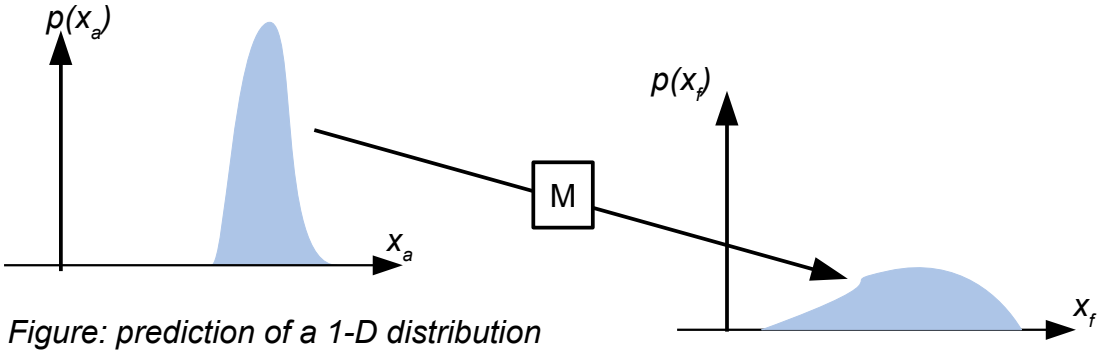


Figure: prediction of a 1-D distribution

Mathematically, the distribution function of x_f is a convolution of the distributions of x_a with the distribution of M. Unfortunately, this is an immensely complex and numerically expensive computation, except for some trivially simple problems. The standard tool for solving it is a numerical integration, but it works very poorly when the dimension of x is large, a problem known as the *curse of dimensionality*. To make a long story short, directly computing uncertainty distributions only works in practice if x has a very small dimension (say, less that 10). For larger-dimension problems, the most effective approach is to approximate the uncertainty distributions by sets of discrete values, a technique known as **Monte Carlo** sampling, also called "**ensemble techniques**" in the ocean and atmosphere communities. This is explained in the next section.

Representing distributions with ensembles

Ensembles are a special family of distributions. Textbook distributions like the gaussian are continuous i.e. the predicted variables can take an infinite set of values. On the other hand, ensembles are finite distributions: the prediction can take one of several values (usually a few dozen, each with the same probability). A mathematician would call it a *Dirac mixture*, that is, a sum of special distributions that put weight on a few values. In practice, we can simply represent an ensemble {x} as a set of values:

$\{x\} = \{x_i\}_{i=1..n}$

where

n is the ensemble size (the number of values)

x_i are the members (the set of values)

if all members have the same probability, which is the most common situation, then each member has weight 1/n. This is called an *equiprobable ensemble*.

The above discussion is best visualized by representing the probability distribution function (PDF) of a real (or *scalar*) value (i.e. of a model state of dimension zero), which is a real function (from R to R). The PDF of a scalar gaussian distribution has the shape of a bell:

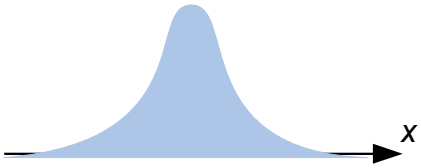


Figure: a gaussian PDF

whereas a rectangular distribution (also called "box car") has a rectangular shape:

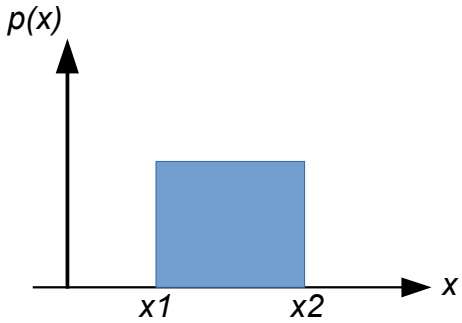


Figure: a rectangular distribution between values x1 and x2

An ensemble of size one has weight on a single value x_1, which would look like a spike (it is a Dirac distribution: strictly speaking, the spike has infinite height and an integral equal to 1):

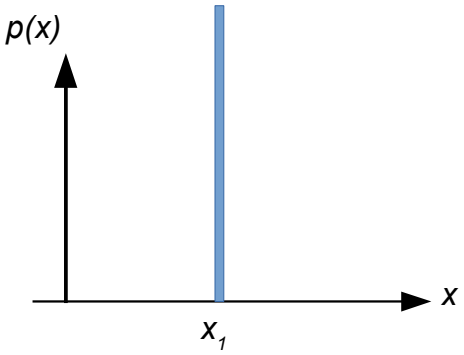
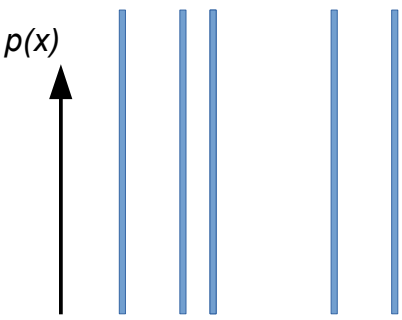


Figure: a Dirac distribution on point x1

A five-member ensemble would look like a kind of comb, with a spike on each value x_i :



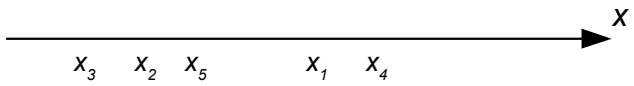


Figure: a five-member ensemble of values

A more readable way of plotting distributions is to show their integral, called the CDF (**cumulative density function**); a probability distribution has an integral of one, so the CDF goes from zero (at -infinity) to one (at +infinity). Here are the CDFs of the four distributions plotted above:

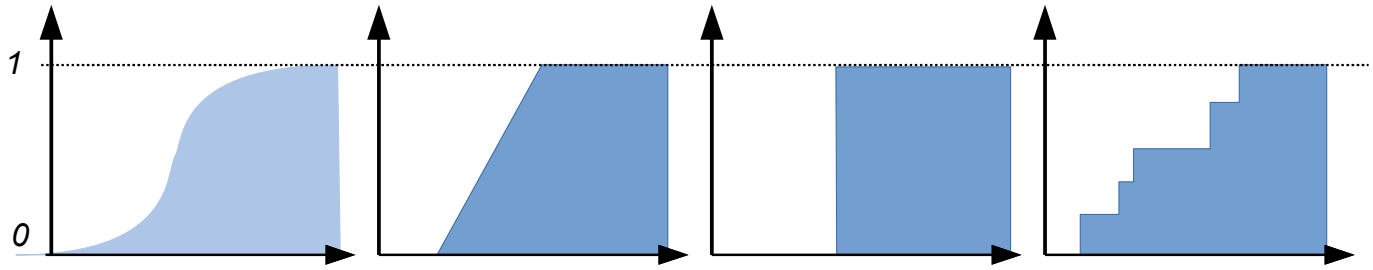


Figure: CDFs of the four preceding distributions

We can see that the CDF of an ensemble has a *staircase shape*:

- the abcissa of each jump is located at the ensemble values x_i
- the CDF value is zero at the left of the smallest value, and one at the right of the largest value
- if there are n members, the height of each jump is 1/n

Note that in the last plot above, the members do not have to be indexed by increasing values. The CDF of an ensemble can only have upward steps, because it is the integral of a positive function.

Non-equiprobable ensembles: in the above example, the ensemble was equiprobable. In some (relatively uncommon) applications, it is desirable to have uneven weights, so that defining the ensemble requires listing the weight wi attached to each member value x_i:

$\{x\} = \{ x_i , w_i \}_{i=1..n}$

An ensemble is a probability distribution, so the sum of weights is always one: $sum_{i=1..n} w_i=x$

The CDF of a non-equiprobable ensemble is a staircase, with uneven steps:

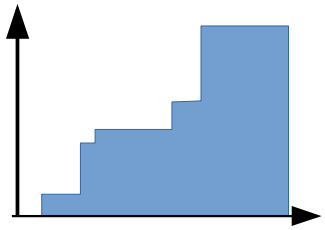


Figure: ensemble CDF with non-equiprobable steps

In most applications, the ensembles are designed to be equiprobable, because this (on average) is the most numerically efficient way of approximating any uncertainty distribution by randomly drawn members. In the following text, all ensembles considered are assumed to be equiprobable.

Basic statistics of a scalar ensemble

In models with many parameters, visualizing all members can be tedious. It is often more efficient to replace the distributions {x} by a few diagnostics, called *summary statistics*, to summarize the general shape of the PDFs. The most useful statistics are defined as follows for an ensemble:

- the ensemble **mean** is the average of the member values: $m(\{x\})=1/n\ sum_{i=1,n}\ x_i$. Geometrically, it indicates a form of centre of the ensemble; its main drawback is its non-robustness: it may be affected by just a few members that are very far from the others.
- the **standard deviation** is the quadratic mean of the distance between the members and the ensemble mean: $sigma(\{x\})=sqrt(\ 1/(n-1)\ sum_{i=1,n}\ (x_i-m(\{x\})^2$. In the above formula, the mean is normalized by n-1 and not n for a rather technical statistical reason, linked to the fact that ensemble values are used twice in the sigma formula, once to compute the mean, once to compute sigma. (If you are not convinced, check from the above formula that for a two-member ensemble, sigma is indeed a measure of distance of the members to the ensemble mean). The standard deviation is a measure of dispersion (= spread), it is very non-robust, because the quadratic mean make it very sensitive to outliers.
- the **median** Q50 is a value such that half of the ensemble values are below it. It is not necessarily unique: for instance, in an ensemble of two different values (x_1,x_2), any value in $[x_1,x_2]$ fulfills the condition. It is usual to take $(x_1+x_2)/2$ as the median, but other conventions may be used. This non-unicity problem can occur in ensemble with larger sizes. The median is a more robust indicator of the ensemble centre, but it can be more expensive to compute, since it requires sorting the members.
- the **quantile** (or *percentile*) of level n%, denoted Qn, is a value of x such that n% of the ensemble values are below it. Thus, the median Q50 is the quantile of level 50%. Quantiles have the same non-uniqueness and computational issues as the median, and they are relatively robust, except for very small or large values of n. By convention, quantiles Q0 and Q100 (also denoted Qmin and Qmax) are often set to the min and max value of the ensemble. Obviously, Qmin and Qmax are extremely

non-robust statistics, and as explained below it is very wrong to think that they represent the min and max possible values of x.

- the ensemble probability of exceeding a threshold t, denoted P(x>t), is the proportion of members with a value x<t. For instance, the *ensemble probabilities* of exceeding Qmin and Qmax are, respectively, 0% and 100%. The ensemble probability of exceeding the median is 50%. More generally, the probability of exceeding quantile Qn is 100-n percent: exceedance probabilities and quantiles are essentially the same thing, but
  - probabilities are uniquely defined, quantiles are not
  - quantiles are expressed in the same physical unit as x, probabilities are not.
  - probabilities can be cheaper to compute, since one only needs to count the members above or below t, sorting them is not necessary.

A related, more subtle difference is that a map of probabilities only indicates how x is positioned with respect to t; it does not matter if x takes very large or very small values above t, so threshold probabilities are not suitable for comparing the tails of distributions with very variable amplitudes (if signal processing terms, they compress the dynamics of the ensemble). Probabilities can get stuck at 0% or 100% levels if the threshold is poorly chosen. Quantiles, on the other hand, tend to 'naturally' follow the dynamics of the ensemble distribution.

The following figure shows how probabilities and quantiles are graphically related to a "staircase" ensemble CDF.

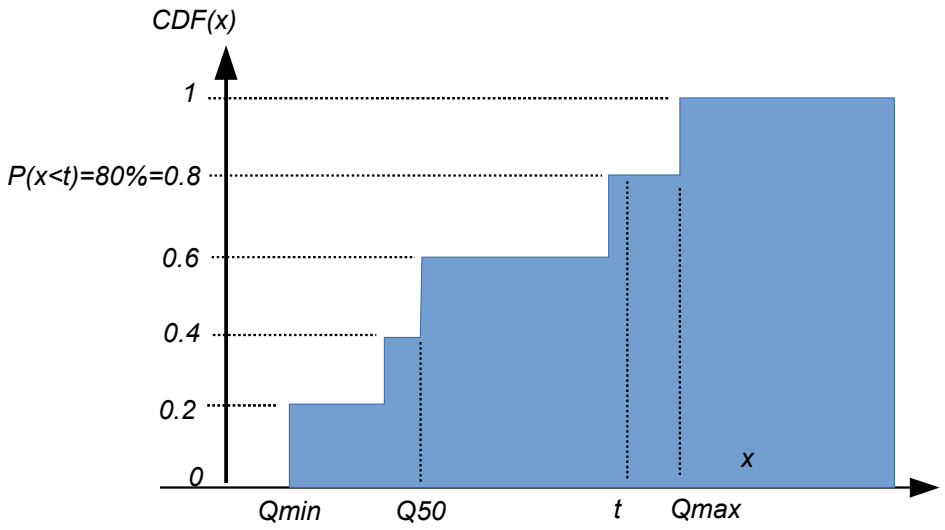


Figure: statistics on an ensemble CDF

Do ensembles represent anything but themselves ?

A frequent argument between ensemble practitioners and theoreticians is the question of whether the CDF of an ensemble really is the above staircase, or it is an approximation of some more complex distribution. This will be illustrated by the following examples:

- case A: assume that we know from some theory that the uncertainty on a scalar variable x has a gaussian distribution N(m,s), but we do not know its mean m or standard deviation s. So we draw a 10-member ensemble {x} using a random number generator that will, in the limit of an infinite-size ensemble, distribute the members as a gaussian N(m,s). The ensemble mean m({x}) and standard deviations s({x})(called the sample statistics) will be some approximations of m and s, with some sampling error. So an engineer might look at the ensemble and say, "the ensemble indicates that x has mean m({x}) and standard deviation s({x})" and the theoretician will (rightly) reply "you are wrong, these are only approximations to the mean and standard deviation of x".
- case B: assume we do not know anything about the distribution of x, except that a 10-member ensemble ({x}) is the best approximation we have. So an engineer will say "the ensemble indicates that x has mean m({x}) and standard deviation s({x})", and when theoretician says "no, these are only approximations", she will reply "these are not approximations, but the exact mean and standard deviation of my distribution. Prove me wrong.". Perhaps the theoretician will try to fit a gaussian or some other smooth, textbook distribution function (gaussian, Weibull, lognormal...) to model the ensemble CDF, and come up with big equations about the sampling error of a 10-member ensemble drawn from it. But it does not matter: here, the ensemble CDF *is* the member staircase. Replacing it with a more complex distribution only adds noise to the CDF, unless one can prove that the actual CDF (that would be obtained with an infinite-size ensemble) has a specific shape. The engineer believes that the textbook distributions are approximations to the ensemble, the theoretician believes the opposite, and there is no data to prove which is right or wrong.

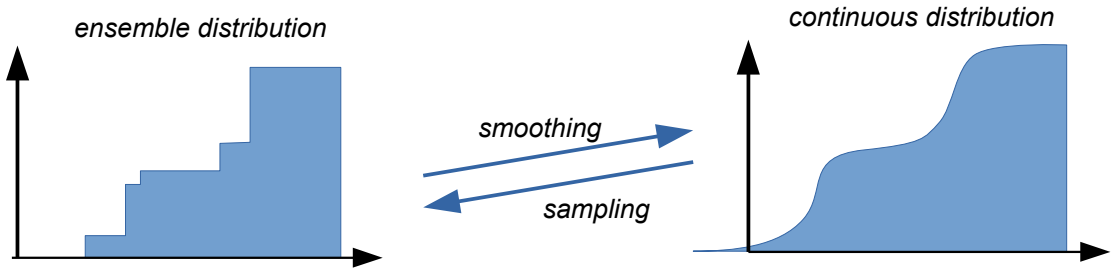


Figure: duality of continuous vs ensemble CDF

As shown by the above discussion, engineers and theoreticians disagree because they base their work on different hypotheses. Textbook probability distributions may be beautiful, but it is rarely possible to prove that they work better than ensemble

distributions, since they are a trade-off between CDF smoothness and faithfulness to the actual values of ensemble members (a deeper reason is that it takes massive amounts of data to prove that a distribution function is better than another, and it is unnecessary to answer such a difficult question in practical problems where one only needs to predict a few statistics, not a complete distribution function). In most real-world problems, CDFs do not have provable shapes, so finite-size samples are the best tool we have to approximate reality.

A few years ago, theoretical statistics used to be useful to avoid doing expensive numerical computations, but increasing computing power now means that in most cases, empirical statistics based on discrete samples are very hard to beat: they save manpower by avoiding complex mathematical derivations. The main cases where theoretical statistics still tend to be useful is when sampling errors are very large, namely, very small ensembles (sizes<10) and the computation of probabilities/quantiles in the distribution tails above Qmax (e.g. for predicting extreme events).

In summary, although it is possible to interpret ensembles as approximations of smooth distributions, it is usually not possible to prove that it works better than using raw ensemble members.

Building random ensemble members

As suggested above, the most common way to build an ensemble of real numbers is to model the uncertainty using a simple distribution, and to draw members using a random number generator.

Using random number generators

Modern computers offer 'random' functions that compute numbers uniformly distributed between 0 and 1, so n draws from such a function u() effectively generate an n-member ensemble $\{u_i\}$ that approximates a uniform (i.e. rectangular) distribution over the [0,1] interval. Any other distribution F can be approximated by applying a well-chosen function g to a uniform distribution, so that a corresponding ensemble is obtained by computing $\{g(u_i)\}$ (a solution is to build g as the inverse of the CDF of F, but there are many others). Languages like python (library *random*) make this even easier, by supplying function that draw directly from the most commonly used distributions (gaussian, weibull, beta, gamma...).

```
import random
for i in range(5) : print( random.gauss(0.,1.) )
...
-0.06572257944854769
-1.1163932186582022
-0.7644811008787148
-0.6909782568792042
-0.48034101267428925
```

Figure: generating a 5-member gaussian ensemble with python3

Using quadrature points

If the model dimension is small and the desired distribution is known, random 'Monte Carlo' draws may not be the most efficient ensemble generation method: most ensemble statistics like the mean, standard deviation, exceedance probabillity, etc are integrals of the ensemble PDF, and their ensemble versions are just numerical integrations:

- the mean of a distribution f(x) is the integral of xf(x)
- its probability of exceeding threshold t is the integral of f over [t,+infinity[
- its standard deviation is a function of its variance, which is the integral of $x^2f(x)$
- etc.

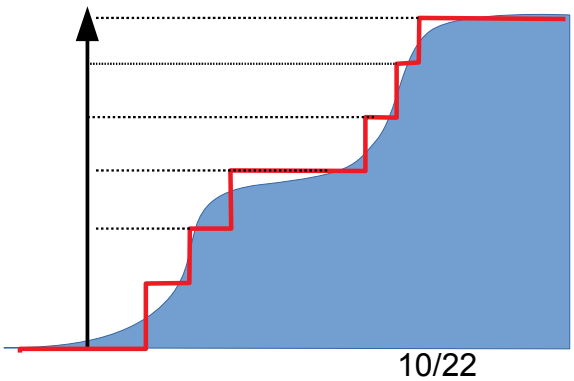


Figure: numerical quadrature of a function, using 6 equally weighted sample points

Thus, asking for an ensemble to be "the best approximation to a PDF" is, from a practical point of view, the same as asking for its statistics to be the best possible approximations of the above numerical integrals, using only the function values at points x_i. There is a vast mathematical literature on how to compute an integral using function values at a fixed number of points x_i, called *quadrature* points. The most important idea is that the quadrature points should be as evenly spaced as possible, in some sense that depends on the properties of f. If x is one-dimensional and the ensemble has size n, a good approach is for the x_i's to sample evenly spaced probability levels:

- if n=1, x_1 should be the median of f
- if n=2, (x_1,x_2) should be the terciles i.e. quantiles Q33 and Q67
- if n=3, (x_1,x_2,x_3) should the the quartiles i.e. quantiles (Q25,Q50,Q75)
- etc.

This idea that ensemble members are even slices of a distribution is very important for practical uses of ensembles, as will be discussed in another section.

If dim(x) is small but >1, there are various algorithms to sample the parameter space in a self-avoiding way: the Halton sequence, sigma points (also known as the unscented transform), etc., each with subtle pros and cons. In high dimensions, it is hard to beat a naive Monte Carlo sampling (i.e. drawing each component of vector x in a random, uncorrelated way).

Generating correlated multidimensional ensembles

Most the above discussion is relevant for model states (i.e. vectors x) that have small dimensions (less a a few dozens). In modern geophysical fluid modelling, model states typically represent fields on 2D or 3D grids of sizes from 10^3 to 10^9 and more. When initializing an ensemble of such model states, a naive strategy is to draw independent random numbers at each gridpoint (mathematically, this is called *white noise*). This is rarely acceptable, because fluids tend to have some spatial smoothness: variations of a physical parameter at one point (temperature, current, salinity, pressure, etc) are nearly always related to some extent with the variations at the neighbouring points. It means that white noise is a poor statistical model, in the sense that fields generated that way have zero probability of occurring in nature. White noise is also likely to generate unphysical behaviour when used to define initial conditions or boundary forcing in numerical prediction models of the environmental fluids (see a data assimilation course for more explanation). Ensemble predictions using this kind of input ensembles will usually lose spread very quickly, meaning that their predictions will be very unefficient representations of the actual forecast uncertainty (see below for the relation between spread and uncertainty). It is much more realistic and effective to create ensemble fields that have spatial correlations over distances similar to the actual forecast errors (spatial correlation and smoothness are two closely related properties).

A relatively simple and cheap way of generating a random, spatially correlated field is the following:

- independently draw a random number at each grid point: it creates a white noise field
- apply a spatial smoothing operator over that field. A simple operator is the Laplacian (a kind of local average), which can be applied several times to increase the correlation length. One can also alternatively apply recursive filters, or spectral filters (i.e. amplitude modulation on the field's energy spectrum), which can be cheaper in some settings. These operators can be configured to produce a predefined correlation length-scale (the typical distance over which spatial correlations are significant)
- renormalize (i.e. rescale) the obtained field so that it has the required mean and standard deviation
- These operations can be redone several times to generate the fields of several ensemble members.

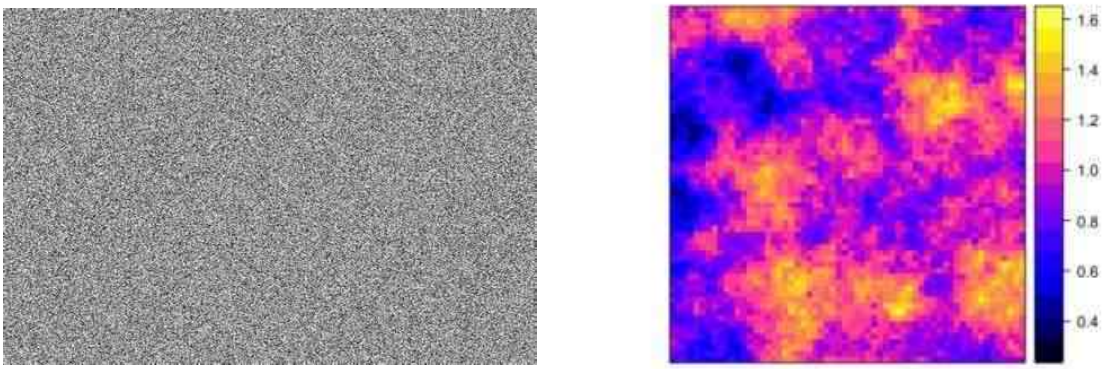


Figure: random fields: white noise (left) and autocorrelated (right)

This kind of algorithm is widely used in meteorology and oceanography, but it does not offer much control over the resulting distribution at each gridpoint. It is a mathematical fact (the 'central limit theorem') that summing several random numbers will usually lead to distributions that are close to gaussian, so unless special treatment is applied to these kinds of ensembles, they will usually look rather gaussian which is ok for most geophysical variables (e.g. temperature or wind) but not others (e.g. chemical concentrations, wind speed, rain or relative humidity). The spatial correlations will tend to be gaussian, homogeneous and isotropic, which may be ok in the horizontal in the free atmosphere, ocean or in well-mixed turbulent flows, but not in boundary layers, or near coasts or fronts, for instance.

Generating ensembles with known covariances

The previous section explained how to generate large model states with a known correlation length-scale. If the model state dimension is not too large, and if one accepts

that the distribution will be gaussian at each gridpoint, it is possible to completely specify the ensemble variances and correlations between all gridpoints, as follows:

- build the required autocovariance matrix M=cov(x,x). For instance, one can specifiy a variance field, and a spatial correlation model to compute the coefficients of M.
- draw n vectors z_i whose components have each some independent gaussian distributions N(0,1) (i.e. mean=zero and standard deviation=1).
- compute a symmetric square root on M, i.e. a matrix N such that $N\ N^T = M$
- build the ensemble $\{x_i\}$ as $x_i = N(z_i)$
- it has the required covariance because $\text{mean}(x_i\ x_i^T) = N\ \text{mean}(z_i\ z_i^T)\ N^T = N\ I\ N = M$ where I is the identity matrix.

This is a powerful method for approximating large covariance matrices with ensembles, used in most Ensemble Kalman Filter algorithms for data assimilation (see the data assimilation course for more detail).

Designing and running ensemble predictions

The previous sections explained how an ensemble of states {x} can represent the uncertainty of a model state x. Now we shall see how to estimate the uncertainty of a numerical forecast represented by a prediction operator M that maps x at present time to another model state, y, at some future time: if x_0 is our best estimate of the current model state, then our best deterministic forecast is

$y_0 = M(x_0)$

that is, if we assume that M is our best available prediction model. Conceptually, we are producing the output data y_0 as a function of our knowledge of x_0 and M, which are our input data:

$y_0 = f(M,x0)$

the uncertainties on the design of M and x imply forecast uncertainties on y. If we use p() to denote the probability distribution of each, we have:

$p(y) = f(p(M),p(x))$

which means "randomly draw all possible values of M and x, compute M(x), the output will be distributed like the uncertainty on y". Unfortunately this is hugely expensive to compute if x and/or M can take infinitely different values (mathematically, f represents the convolution of p(x) by p(M); if x and M are Dirac distributions, then f(M,x) equals M(x) which is the deterministic forecast above). We can approximate p(y) b an ensemble of n members as follows:

- draw n random values of x, $\{x_i\}$, to approximate p(x)
- independently draw n random values of M, $\{M_i\}$, to approximate p(M)
- then, the set of $y_i=f(M_i,x_i)=M_i(x_i)$ is an approximation of the forecast uncertainty y_i called an **ensemble prediction.**

[Figure: schematics of ensemble perturbation & output]

The exact proof of the above is a bit complex, but it boild down to saying "to estimate the uncertainty of a forecast, perturb all uncertain input parameters of the forecast process". The subtelty is that one should in principle take into account the uncertainties in everything, including the forecast model equations M, including all processes that exist in nature but are not represented by M. Doing so is obviously hopeless since we cannot represent the uncertainty of things we do not know about. Fortunately, in practice we "only" need to represent the uncertainty on processes to which the forecast is significantly sensitive. Also, as will be shown below, we can account (to a limited extent) for error sources we may have forgotten by doing some machine learning.

Concrete example: modern global numerical weather predictions are known from experience to be significantly affected by the following uncertainties:

- errors in the model initial condition (because we do not have enough observations to perfectly known the current weather everywhere on the globe)
- errors in modelling slowly evolving parameters of the lower boundary of the atmosphere, like sea surface temperature (SST), soil moisture, etc
- errors in the model forecast equations, in particular because the model resolution is not infinite (global models have horizontal grids >1km wheras the atmosphere has

significant motion like eddies down to 1mm size) and the computation of some physical processes is a poor approximation to their real behaviour (e.g. the thermodynamics of convective clouds, the rain patterns, the radiative effect of clouds)

Thus, an ensemble prediction system to estimate the forecast uncertainty of the above procedure could be:

- draw n independent perturbations to the best available analysis, using for all parameters the standard deviations and correlations that are known (from statistics on past analyses) to to occur on average in the analyses: this yields an ensemble xi of initial conditions. (the data assimilation course explains more effective ways of representing analysis uncertainty, using *Kalman filters* or *ensemble data assimilation*), because the amplitude and correlations of analysis errors are not the same everyday or everywhere, and it is possible to predict their variations)
- draw n independent perturbations to surface fields of SST, soil moisture, etc, again using statistics gathered from historical error diagnostics
- draw n modifications to the equations of the numerical prediction model M, to represent known errors linked to the representation of small eddies, convective clouds, radiation, etc. the combination of these perturbations with the set of initial conditions defines a set of n forecast model M_i
- apply these n model and surface condition variants to the n initial conditions: it yields an ensemble representation $\{y_i\}= \{ M_i(x_i) \}$ of the forecast uncertainties on y.

By "perturbations" we mean that random numbers are added to some deterministic prediction system to represent an ensemble of prediction systems, which is the most common ensemble generation technique.

Multimodel ensembles: in some prediction systems, ensembles can be obtained directly by gathering fields and model software from different, independent prediction system: this is called the *multimodel* technique, in which the predicted uncertainty is not linked to a particular model, it represent the knowledge gathered from the set of models used. (multimodel ensembles are a convincing way of representing model error, unfortunately they represent a lot of effort to run since one needs either to master many different prediction software systems, or to gather predictions in real time from many different weather prediction centres. They also raise statistical difficulties linked to varying systematic errors beteen the ensemble members, which creates an ensemble dispersion that is not truly "random").

Tuning & measuring the quality of an ensemble prediction

The above explains the mechanics of generating an ensemble prediction. But how do we tune the ensemble perturbations ? How can we optimize the quality of an ensemble prediction system ? Because a probabilistic prediction produces not a value but a function of every forecast parameter, there is an infinity of ways to define and measure the quality of en ensemble forecast. Unlike a deterministic forecast where one can just measure a distance between a forecast value and a truth (e.g. observed) value, the most important concept to understand is that

***** It is impossible to judge the value of an ensemble forecast on a single case *****

(sadly, this is still not understood by many ensemble users, even after over 30 years of operational ensemble prediction in meteorology). Because probability distributions can only be accessed through statistics like spread, probabilities, quantiles, etc, ensembles can only be evaluated on average, on samples large enough to compute these statistics at the precision needed to reach practical conclusions. Conclusions made on isolated cases are usually misleading.

The most robust statistics (i.e. the easiest to compute with a small sample) are the ensemble mean and the spread.

Step one: validating the ensemble mean

There are three kinds of important mean-type statistics to check with respect to observations:

- the **ensemble mean minus the observation mean** is equal to the mean of (member - obs) values i.e. it is the ensemble bias. It can be used to study if an ensemble prediction system has biases: because numerical models are non-linear, even symmetric perturbations distributions can create forecast biases, which may or may not have predictive value. If the ensemble members are *interchangeable* (i.e. they are all drawn from the same, uncorrelated distributions of perturbations), the ensemble mean is not very interesting because it contains the same information as can be obtained from a single member i.e. a deterministic forecast. Actually, a "deterministic" numerical prediction started from an unperturbed variational or Kalman filter analysis *is* an ensemble mean in the sense that it started from the mean of the analysis uncertinty distribution (see the data assimilation course). Multiphysics or multimodel ensemble members are usually NOT interchangeable, so there is a complex relationship between the members biases and the ensemble spread.
- the **distance between the ensemble mean and the observations** (e.g. their absolute difference or the quadratic distance) measures how far the ensemble centre is from the truth. It is popular with some ensemble developers because it is often smaller than the score of the corresponding deterministic forecast (obtained by setting all ensemble perturbations to zero), so they can tell their managers "look, my ensemble predictions are better than the deterministic system". Taking the mean produces this apparent improvement when it hides some atmospheric (or oceanic) variability, by averaging out the differences between members and only retaining the common flow features. The result is that the ensemble is generally a useless forecast because it lacks many features present in the members (mathematically,

the reduction in the quadratic distance is linked to the "parallel axis theorem"). In conclusion, this should never be used as an ensemble score, except as part of the spread-skill diagnostic explained in the next section.

- the **mean of the distance between the members and the observations** (notice how it differs from the previous measures) measures how far each member is from the truth. It is usually larger than the equivalent score of the corresponding deterministic forecast, because members are, to first order, equal to the deterministic forecast plus some random uncorrelated noise. It is usually necessary to degrade this "distance" (with respect to the deterministic forecast) in order to create a realistic ensemble spread (due to forecast nonlinearities, sometimes this distance is insensitive or even decreases with ensemble spread, for instance with some stochastic physics schemes). Thus, it is NOT a standalone measure of ensemble quality. It is a common mistake in ensemble verification to conclude that an ensemble is "worse" than another model because its members are further away from the truth according to this kind of diagnostic. It is interesting, however, for checking the impact of nonlinearities in an ensemble prediction, by investigating the question "does adding random noise to the output of a deterministic prediction produce better or worse forecasts than perturbing the inputs of the corresponding ensemble prediction ?"

[Figure: score rms & bias]

Step two: validating the ensemble spread

The spread (or 'dispersion') measures a distance between the ensemble members and an ensemble centre. The most common measure of spread is the standard deviation. Achieving an ideal value of an ensemble spread is a major goal of ensemble design, because it is closely related to the design of ensemble perturbations:

- an "ensemble" with no input perturbations will have zero spread.
- ensemble spread usually increases (more or less linearly) with the standard deviations of input perturbations.
- ensemble spread is a direct measure of forecast uncertainty.

Standard deviation is a poor indicator of uncertainty for variables with very non-gaussian distributions like precipitation, cloudiness or concentration, but it is a useful indicator of the correctness of ensemble spread when verifying with quasi-gaussian variables like temperature and wind or current components (but not speed). More precisely, an ensemble with insufficient spread:

- has members and quantiles that are, on average, too close to the ensemble centre and to the corresponding deterministic forecast (i.e. it lacks variability in the predicted scenarios)
- has differences between members that are, on average, smaller than the differences between the members and reality (we say that "reality escapes the ensemble")

- scalar forecast parameters are too often observed to be below the ensemble min or above the ensemble max (although it may happen for a perfectly spread ensemble, it should not happen too often: see the section on reliability below)
- its probabilities are, on average, too close to zero or to one (we say that "the ensemble is overconfident")
- in summary, an insufficiently spread ensemble underestimates the forecast uncertainty

The correctness of spread is a particular measure of ensemble reliability (see next section), it is particularly important when using ensembles to estimate forecast errors,but as can be seen from the above list, lack of spread pollutes most aspects of ensemble output. Achieving enough spread is a major concern in most ensemble prediction systems, which tend to be underdispersive, so that many ensemble developers are looking for ways to increase their spread. One needs to be careful not to exaggerate ensemble spread, because many ensemble scores can hide the following issues:

- in a single ensemble, some output parameters can be overspread, while others are underspread.
- averaging spread over a large data samples can hide compensation between flow regimes that are underspread, and others that are overspread.

The two most commonly used measures of spread correctness are the rank diagram (described in the next section) and the spread-skill ratio, which is computed as follows: for a scalar forecast parameter ensemble {xi} and its observation xo, with an observation error standard deviation so:

- let s(xi) be the forecast standard deviation at one particular point
- let m(xi) be the ensemble mean
- then spsk(xi,o) = ($s(x_i)^2 + s_o^2$) / sqrt($(m(x_i)-x_o)^2$) is called the **spread-skill ratio**

It can be demonstrated that an ensemble with a perfect forecast distribution has, on average, spsk(x_i,o)=1. If it is <1, then the ensemble is underspread.

[Figure: underspread vs overspread ensemble]

Note that the first term of the denominator of spsk() is the quadratic distance between the ensemble mean and the observation. Also note that observation errors act as if they *increase* the ensemble spread, wheras intuition might have suggested it is the other way around. The condition spsk()=1 can be rephrased in English as follows:

From the point of view of quadratic distance, (member,member) couples should be undistinguishable from (member,obs) couples.

This is a particular case of reliability, as explained in the next section.

A common problem with spread tuning is when we only observe a small fraction of model variables. Inflating the spread for those variables that are observed does not guarantee that spread in the tuned ensemble will be correct for all variables. Many ensemble perturbation techniques generate substantial correlations between variables, because they rely on some aspects of model design. It can even lead to some perturbations schemes

creating random large-scale biases (e.g. tuning parameters in physical parametrizations, or multiphysics), which means that although the spread measured and tuned at observation points (assumed to be uncorrelated between points in space and time) may be correct, there may be a large overperturbation of large-scale/low-frequency structures. This problem is colloquially known as "when there is a hammer in your hand, everything looks like nails".

[Figure: overspread ensemble by partial tuning]

Step three: validating the ensemble reliability

The above criteria measured some form of consistency between the ensemble mean and spread, and the observation. A more general form of consistency, called reliability, generalizes these measures as follows:

The distributions of observations and ensemble members should be undistinguishable from each other.

Note that reliability is a necessary condition for an ensemble to prodict the correct uncertainty distributions. It is not sufficient.

Apart from the ensemble mean and spread, the most frequently used measures of reliability are, for scalar variables:

- the *reliability diagram*, which checks if probabilities of exceeding one threshold are consistent with the observations.
- the *rank diagram*, which checks if the relative ordering of members and observations can be distinguished from each other.

Calibration is a set of post-processing techniques by which some aspects fo the reliability of an ensemble are improved using machine learning.

[Figure: reliability diagram]

The rank diagram is the visualization of a fundamental property of the ensemble prediction of a scalar variable:

when sorted in ascending order, the values of an n-member ensemble define n+1 classes of equally likely values. The probability of the truth being between Qmin and Qmax is (n-1)/(n+1).

A sadly often forgotten consequence is that there is 2/(n+1) probability that truth "escapes the ensemble" i.e. is under Qmin or above Qmax. For a 12-member ensemble, this occurs 2/13= 15% of the time ~ every 7 cases. Some weather forecasters fool themselves into thinking the ranges of values predicted by ensembles are bounds what can possibly happen, sometimes with disastrous consequences. Interpreting ensembles that way requires computing them with enough members (n at least 30 to 50) so that the ratio 2/(n+1) becomes negligible (less than the tolerated non-detection rate), which is a computational investment that few forecasting centres have done so far. This problem explains most current failures to successfully apply ensembles as tools for human forecasting.

[Figure: rank diagram]

<to be written>

The concepts of reliability and resolution

Reliability is a "climatological" measure of the correctness of an ensemble, by checking that average predicted distributions are consistent with various aspects of the observed distributions. Reliability is a necessary, but not sufficient condition for an ensemble to correctly predict the forecast uncertainty. An ensemble can have perfect reliability according to many measures, by producing a perfect climate, but no usefulness, by randomly issuing climatologically distributed values.

A dual aspect of ensemble performance is the *resolution*, i.e. the ability of an ensemble to predict probabilities that are substantially different from climate (do not confuse *ensemble resolution*, which is a statistical property, with *model numerical resolution* which is related to grid discretization). Resolution is not a matter of consistency between ensemble and observation, it just measures whether probabilities are rather "iffy" (close to some average value) or rather categorical (close to zero or one): it does not depend on observations. An ensemble can have perfect resolution by only producing probabilities equal to zero or one, and be useless if these values are unrelated to what happens in reality. The CRPS and Brier scores (explained below) can be decomposed as the sum of a reliability term and a resolution term.

It can be shown that an ensemble that has perfect resolution AND perfect reliability produces perfect forecasts, by issuing probabilities equal to 1 if and only if the corresponding event occurs. It means that there is no forecast uncertainty, in which case it is useless to compute an ensemble since any member is a perfect deterministic forecast. Real-world prediction systems have imperfect forecasts so they cannot simultaneously have perfect resolution and reliability. The difficulty is understanding the difference between "perfect forecast" and "perfect ensemble":

- a "perfect forecast" makes predictions that are equal to observations (within observation error). It is deterministic, since by definition it has is no uncertainty.
- a "perfect ensemble" is an ensemble that correctly represents the forecast uncertainty of a given prediction system. It the latter is imperfect, then the ensemble cannot have perfect resolution and reliability.

Confusion between the concepts of "good prediction system" and "good ensemble" is a frequent cause for misunderstanding. A correct way of dealing with it is to only use scores to compare between pairs of prediction systems, or to directly look at the user value.

Ensemble calibration may or may not improve ensemble resolution and/or user value.

Step four: validating the ensemble user value

User value is the correctness of the decision a user can make using information from an ensemble. It depends on the ensemble and on the kind of information that is beneficial (or harmful) to the user, i.e. the *user impact*. Different users may experience different ensemble values.

Literally hundred of other ensemble scores have been proposed over the years, each author usually claiming their score is better than the others. Of course, it is a matter of point of view, since there are infininte degrees of freedom in an ensemble forecast. For instance some authors have advocated for scores to be "proper", "fair", "symmetric",

related to "entropy", etc, concepts which have little or no meaning in practice. What matters in the end is what value the users are able to extract from ensembles, which requires designing ensemble scores with the user's interests in mind. A nice consequence is that the variety of scores to consider inevitably boils down to the variety of user sensitivities that are worth considering. We will explain only four such scores, and let it to the reader to invent their own scores if they need to target other uses.

Note that all scores presented here are scalar (i.e. they score 0-D, real variables). They are not ideal for validating higher-dimensional aspects of the ensemble predictions, such as relationships between parameters, field structures (which requires "object-oriented" tools) and forecast scenarios.

The **CRPS score** (continuous ranked probability score) is sensitive to the overall shape of the uncertainty distributions. Although its simplicity may seem attractive (a single score acts on the whole distribution), it is at the same time a problem, because it makes the CRPS sensitive to all parameter thresholds, whereas the user may only be interested in some of them. The CRPS also tends to be oversensitive to large-scale aspects of the reliability (e.g. the ensemble bias and standard deviation) at the expense of other aspects of the distributions that may be important, like the probabilities of extremes. The CRPS can be decomposed into the sum of a score that only measures the reliability and the resolution of the forecast.

[Figure: CRPS conceptual diagram]

The **Brier score** is a simplification of the CRPS that only looks at distributions in terms of the exceedance of a probability threshold (indeed, one can show that the CRPS is an integral of the Brier over a range of thresholds). Its advantage (and drawback) is that it is insensitive to any other aspect of the forecast errors.

CRPS and Brier can be used as measures of user value if one understands what they mean. They are mostly useful for comparing pairs of ensembles, since the concept of a "perfect forecast", as explained in the previous section, is ambiguous.

The **ROC** (relative operating characteristic) diagram and ROCA (ROC area under the curve) scores take a dual approach from CRPS and Brier: instead of measuring "what is the distribution's worth if one takes a given point of view", they measure "what is the value of decisions made by using the distribution in many different ways". The ROCA is computed from a function called "the ROC diagram", which is a parametric curve:

- set a fixed probability threshold a. The ROC will score probabilities P(x>y) of exceeding y. y has the physical unit of predicted parameter x.
- let a parameter t (called decision threshold) vary between 0 and 1
- for all verification points ($\{x_i\}$,xo),
  - compute the forecast probability >t (typically, by counting the number of ensemble members that have value >y)
  - if $P(x_i>y)>t$, we say that event x>a is "forecasted yes"; if the observation value xo>y we say the we event is "observed yes"; (and 'no' in the other cases). Then we put this verification point into one of 4 categories:
  - category 'a' = 'correct yes forecast' if 'forecasted & observed yes'
  - category 'b' = 'false alarm' if 'forecasted yes and observed no'
  - category 'c' = 'non-detection' if 'forecasted no and observed yes'

    - category 'd' = 'correct no forecast' if 'forecasdted no and observed no'
- we set (a,b,c,d) to the number of verification points that fall in each category.
- ratio POD(t)=a/(a+c) is called "probability of detection"
- ratio FAR(f)=b(b+d) is called "false alarm rate" (note: some authors have a different definition for FAR. Here, the denominators do not depend on the forecasts)

The curve made of points (FAR(t) , POD(t)) is called the ROC diagram. The higher the curve, the better the user value. The ROCA is the integral of the ROC i.e. the area under the curve.

[Figure: ROC diagram and its unfolding]

It can be shown that

- the ROCA is always convex, starts at (0,0) and ends at (1,1), otherwise the prediction system can be improved by making trivial changes to its output
- a ROCA=1/2 i.e. a ROC curve on the diagonal can be obtained by making random i.e. worthless predictions
- if the ROCA reaches point (0,1) (the top left corner), this point is a perfect forecasting system
- the Brier score is an integral of the ROCA
- the ROC and ROCA are not changed by ensemble calibration.

The economic value score

The "potential economic value" diagram, often nicknamed "economic value", is a transform of the ROC curve that makes it easier to relate to the user aversion for false alarms and non-detections. Be aware that the economic value diagram only shows the potential value of a perfectly calibrated ensemble. If the ensemble is not reliable, he/she will not be see this value, but a lower one.

<to be written>

[Figure: economic value diagram & probability]