Training and Calibration

Josef Teichmann

ETH Zürich

Vienna Summerschool 2020

Introduction

Introduction

Goal of this talk is ...

- to understand training as a calibration problem.
- to highlight on some aspects in training procedures.
- to show some applications to model calibration in finance.

(joint works with Christa Cuchiero, Matteo Gambara, Wahid Khosrawi-Sardroudi)

Introduction

Goal of this talk is ...

- to understand training as a calibration problem.
- to highlight on some aspects in training procedures.
- to show some applications to model calibration in finance.

(joint works with Christa Cuchiero, Matteo Gambara, Wahid Khosrawi-Sardroudi)

Introduction

Goal of this talk is ...

- to understand training as a calibration problem.
- to highlight on some aspects in training procedures.
- to show some applications to model calibration in finance.

(joint works with Christa Cuchiero, Matteo Gambara, Wahid Khosrawi-Sardroudi)

Optimization problems and inverse problems

Optimization by integration

We start with a very generic point of view on optimization: consider a non-negative measurable function f on a probability space (Θ, π_0), then calculating the essential supremum of f corresponds to calculating its ∞-norm, for which a well-known formula exists

$$\lim_{n \to \infty} ||f||_n = ||f||_\infty .$$

Optimization by integration

We can interpret this equality in the case $f = \exp(-L)$ for a finite measurable function L and obtain immediately

$$\text{ess-inf}_{\theta\in\Theta} L(\theta) = -\lim_{n\to\infty} \frac{1}{n} \log\Big(\int \exp\big(-nL(\theta)\big)\pi_0(d\theta)\Big) \quad (EQ)\,.$$

Laplace principle

Assume additionally that L is bounded from below. Then we expect the integrand to concentrate at arguments where values close to infima are taken, in other words it is worth investigating the probability measure

$$\pi_n(d\theta) := \frac{\exp\big(-nL(\theta)\big)\pi_0(d\theta)}{\int \exp\big(-nL(\theta)\big)\pi_0(d\theta)}\,.$$

Concentration around arguments where values close to infima are taken can be interpreted by proving that

$$\lim_{n\to\infty} \pi_n[A] = 0$$

for all measurable sets A such that there exists $\epsilon > 0$ and $L(\theta) > \text{ess-inf}_{\theta\in\Theta} L(\theta) + \epsilon$ for $\theta \in A$.

Bayesian optimization

Summing up this yields the following statement: for a measurable function L essentially bounded from below the measure π_n concentrates at arguments where values close to the infimum are taken.

This statement has a Bayesian interpretation. Consider π_0 as prior on Θ and consider L a (negative) log-likelihood, then π_1 is the posterior calculated by Bayes formula (when do not have data in the moment), π_n appears as interation of this procedure and concentrates at arguments for the likelihood maximizes.

Bayesian inverse problems

Let us introduce data in the next step, i.e. we consider the function L as a measurable function of two variables z, the data, and θ, the parameter, on a product space $Z \times \Theta$, where Z only has a measurable structure. We shall write $L^z := L(z, .)$ and assume this function to be bounded from below and measurable. Then we can apply the previous considerations in a data-dependent way.

As a remark we add: for fixed $\theta \in \Theta$ we can sometimes view $z \mapsto \exp(-L^z(\theta))$ as density of a random variable Z with respect to some reference measure ν on Z. In this case the fully Bayesian interpretation takes place, but actually we do not need this here.

Inverse problems

A parameter-dependent optimization problem is an inverse problem: we are interested in describing a map

$$z \mapsto \theta^*(z) \in \operatorname{arginf}_{\theta \in \Theta} L^z(\theta)$$

for some $z \in Z$. We give ourselves additionally topologies on Z and Θ with corresponding sigma algebras being the Borel sigma algebras. We can require, following Jacques Hadamard, the following properties of such a map:

1. Existence for a large subset of Z.
2. Uniqueness for a large subset of Z.
3. Stability where it is uniquely defined, i.e. continuity as a map from Z to Θ.

Regularized inverse problem

Usually it is delicate to guarantee the three properties, which, however, are important if $z \in Z$ are considered data and $\theta^*(z) \in \Theta$ a selected model (identifified by a parameter). Often those properties can be achieved if the problem is replaced by a regularized problem by adding a regularization term $P : \Theta \to \mathbb{R}_{>0}$, i.e. we consider

$$\inf_{\theta \in \Theta} L^z(\theta) + \lambda P(\theta)\,,$$

where $\lambda > 0$ is an additional parameter.

This by now classical theory has been developed in many directions, we shall compare it here with the above developed Bayesian perspective. Consider a reference measure ν on Θ and define a prior

$$\pi_0(d\theta) = \frac{\exp(-\lambda P(\theta))\nu(d\theta)}{\int \exp(-\lambda P(\theta))\nu(d\theta)}\,.$$

Then the posterior

$$\pi_n(d\theta) = \frac{\exp(-nL^z(\theta) - \lambda P(\theta))\nu(d\theta)}{\int \exp(-nL^z(\theta) - \lambda P(\theta))\nu(d\theta)}$$

can be considered a generalized solution of the inverse problem (depending on parmaters n and λ), which concentrates at arguments, where values of $L^z + \frac{\lambda}{n}P$ are close it their infimum. Notice that it is relatively easy to guarantee that π_n depends continously on z.

Via regia to calculate optima

The law

$$p_\epsilon(\theta)d\theta := \frac{1}{Z_\epsilon}\exp\big(-\frac{L^z(\theta)}{\epsilon}\big)d\theta\,,$$

where denominator Z_ϵ is just the integral $\int_\Theta \exp(-L^z(\theta)/\epsilon)d\theta$.

The measure $p_\epsilon d\lambda$ is the invariant measure of the stochastic differential equation

$$d\theta_t = -\frac{1}{2}\nabla L^z(\theta_t)dt + \sqrt{\epsilon}dW_t\,,$$

which is just checked by the following equality

$$\int_\Theta \big(-\frac{1}{2}\nabla L^z(\theta)\nabla f(\theta) + \frac{\epsilon}{2}\Delta f(\theta)\big)p_\epsilon(\theta)d\theta = 0$$

for all test functions f.

This can also be seen as a gradient flow on probability measures with respect to the Riemannian structure corresponding to Wasserstein-2-distance (see talk of Lukasz Szpruch for all references).

Subriemannian via regia to calculate Optima

The law

$$p_\epsilon(\theta)d\theta := \frac{1}{Z_\epsilon}\exp\big(-\frac{L^z(\theta)}{\epsilon}\big)d\theta\,,$$

where denominator Z_ϵ is just the integral $\int_\Theta \exp(-L^z(\theta)/\epsilon)d\theta$.

The measure $p_\epsilon d\lambda$ is the invariant measure of the stochastic differential equation

$$d\theta_t = -\frac{1}{2}\sum_{i=1}^{d}(V_i L^z)(\theta_t)V_i(\theta_t)dt + \sqrt{\epsilon}\sum_{i=1}^{d} V_i(\theta_t)\circ dW_t^i\,,$$

where V^i are standard vector fields on a nilpotent Lie group G_d^m (being homeomorphic to some $\mathbb{R}^M$).

This can also be seen as gradient flow on probability measures with respect to the Riemannian structure corresponding to the sub-Riemannian Wasserstein-2-distance (see, e.g., work of Baudoin-Hairer-T).

The free energy

The quantity of which gradients are taken is

$$F(\rho) := \frac{1}{2}\int_\Theta L^z(\theta)\rho(\theta)d\theta + \frac{\epsilon}{2}\int_\Theta \log\rho(\theta)\rho(\theta)d\theta$$

for measures $m(d\theta) = \rho(\theta)d\theta$ absolutely continues with respect to Lebesgue measure. This can also be understood as relative entropy of $\rho d\theta$ with respect to p_ϵ times $\epsilon/2$.

The gradient flow can also be understood in the following way: consider a starting law ρ^0 and solve the recursion

$$\text{arginf}\{\rho \mid \frac{1}{2}d(\rho^k,\rho)^2 + hF(\rho)\}$$

for time intervals of length h (this is the variational version of the gradient flow on Wasserstein-2-space).

Gradients in Wasserstein geometry

The gradient of a function $m \mapsto F(m)$ has to satisfy

$$\frac{d}{ds}|_{s=0} F(m + s\eta) = \sum_i \int_\Theta \mathcal{D}_m^i F(\theta) V_i g(\theta) m(d\theta)$$

with respect to the flat derivative. The signed measure η is a tangent direction for the flat derivative formed from a tangent direction at m, which is given through transport by a flow with vector field $\sum_i V_i g V_i$, i.e.

$$\int \sum_i (V_i g V_i f)(\theta) m(d\theta) = -\int f(\theta) \eta(d\theta)$$

for test functions f and potentials g.

Mean field aspect

Assume that Θ is a convex polish space, in particular for probability measures $m \in \mathcal{P}(\Theta)$ it makes sense to consider the barycenter of m which lies again in Θ.

Having the previous Langevin dynamics in mind, one can consider

$$d\theta_t = -\frac{1}{2}\sum_{i=1}^{d}\left(\mathcal{D}_{\mathsf{law}(\theta_t)}L^z\right)(\theta_t)dt + \sqrt{\epsilon}\sum_{i=1}^{d} V_i(\theta_t)\circ dW_t^i\,,$$

where the law of θ_t appears as an additional quantity in the loss function, which takes into account the behavior of searches depending on the initial value θ_0. $\mathcal{D}_m$ is the Lions derivative with respect to subriemmanian Wasserstein geometry at m.

Again we can understand the solution of this McKean-Vlasov dynamics as a gradient flow on probability measures with respect to corresponding natural structures, or as a limit of particle systems.

Training a feed forward neural network

We assume a feedforward neural network g^θ on $\mathbb{R}^k$ depending on trainable parameters $\theta \in \mathbb{R}^M$ for a supervised learning task to take values $y_i \in \mathbb{R}^I$ at $x_i \in \mathbb{R}^k$, for $i = 1, \ldots, N$. We define a loss function

$$L^z(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left(g^\theta(x_i) - y_i\right)^2 = \frac{1}{N/k} \sum_{i=1}^{N/k} l_i(\theta)\,,$$

where we understand 'data' z as the collection of (x_i, y_i) and write the loss function with minibatches of size k.

The stochastic gradient descent (SDG) algorithm essentially does the following (with respect to a learning rate γ_n)

$$\theta_{n+1} = \theta_n - \gamma_n \nabla l_n(\theta_n)$$

where θ_0 is sampled from an initial distribution and which is relatively early stopped.

At this point it is questionable whether this samples from any of the previous continuous time via regias, in particular since in the previous approaches the initial value does not play a role whereas it does to some extent in the the SDG above.

Implicit regularization: regularizing impact of initial distribution (comparable to classical regularization functionals) and regularizing impact of minibatching (possibly entropic on probability measures).

An exemplary result for implicit regularization

Randomized shallow networks (RSN)

Let $X \subset \mathbb{R}^d$ compact, $\sigma : \mathbb{R} \to \mathbb{R}$ Lipschitz continuous activation function. $\mathcal{RN}_w : X \to \mathbb{R}$ s.t.

$$\mathcal{RN}_w(x) := \sum_{k=1}^{n} w_k \, \sigma \left(b_k + \sum_{j=1}^{d} v_{k,j} x_j \right) \tag{RSN}$$

- $n \in \mathbb{N}$
- $w_k \in \mathbb{R}$, $k = 1, \ldots, n$
- $(b_k, v_k) \sim \mathbb{P}$, i.i.d. $k = 1, \ldots, n$. $\mathbb{P}$ probability measure on $\mathbb{R}^{d+1}$.

Randomized shallow networks (RSN) - II

Corollary 1. Universal in probability

RSNs are universal in probability, i.e. let $f \in C(X, \mathbb{R})$. If $\mathbb{P} \gg \lambda^{d+1}$ then

$$\forall \epsilon \in \mathbb{R}_+, \lim_{n \to \infty} \mathbb{P}^n \left(\exists w \in \mathbb{R}^n : ||\mathcal{RN}_w - f||_\infty > \epsilon \right) = 0.$$

Lemma 2. Almost sure perfect interpolation

Let observations $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, $i = 1, \ldots, N$ be given.For any $\mathbb{P} \ll \lambda^{d+1}$, a perfectly trained RSN with $n \geq N$ hidden nodes almost surely interpolates the data, i.e.

$$\mathbb{P}^n(\exists w \in \mathbb{R}^n : \mathcal{RN}_w(x_i) = y_i, \quad \forall i = 1, \ldots, N) = 1.$$

Randomized shallow networks (RSN) - II

Corollary 1. Universal in probability

RSNs are universal in probability, i.e. let $f \in C(X, \mathbb{R})$. If $\mathbb{P} \gg \lambda^{d+1}$ then

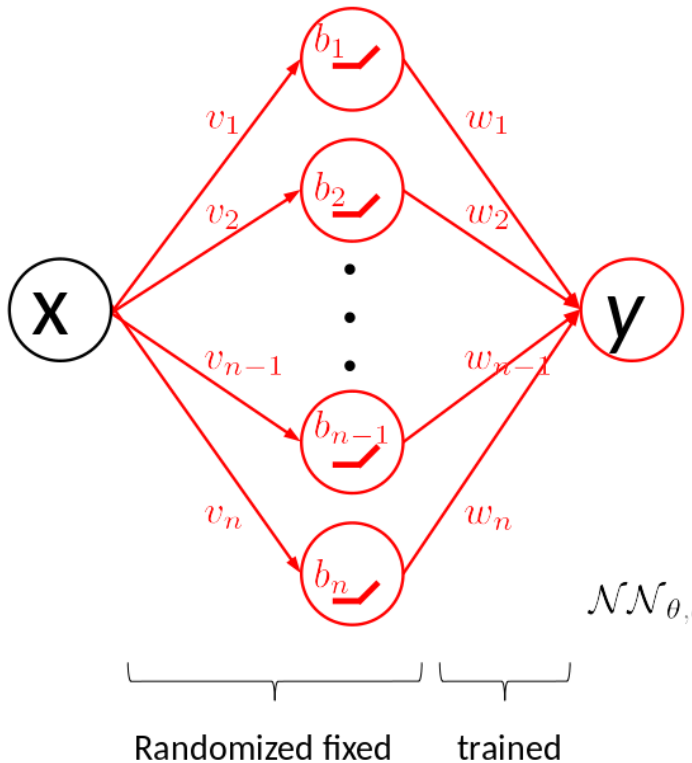
$$\forall \epsilon \in \mathbb{R}_+, \lim_{n \to \infty} \mathbb{P}^n \left(\exists w \in \mathbb{R}^n : ||\mathcal{RN}_w - f||_\infty > \epsilon \right) = 0.$$

Lemma 2. Almost sure perfect interpolation

Let observations $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, $i = 1, \ldots, N$ be given.For any $\mathbb{P} \ll \lambda^{d+1}$, a perfectly trained RSN with $n \geq N$ hidden nodes almost surely interpolates the data, i.e.

$$\mathbb{P}^n(\exists w \in \mathbb{R}^n : \mathcal{RN}_w(x_i) = y_i, \quad \forall i = 1, \ldots, N) = 1.$$

Shallow Randomized Neural Net



$$\hat{f} = \mathcal{NN}_\theta$$

$$\theta = w \in \mathbb{R}^n$$

$$\mathcal{NN}_{\theta,\omega} := \sum_{k=1}^{n} w_k \max\left(0, b_k(\omega) + v_k(\omega)x\right) \quad \forall \omega \in \Omega \; \forall x \in \mathbb{R}$$

RSN and gradient descent - implicit regularization in 1d

Setting

- $D := \{(x_i, y_i) \in \mathbb{R} \times \mathbb{R},\ i = 1, \dots, N\}$
- $\mathcal{RN}_w : \mathbb{R} \to \mathbb{R}$, $\mathcal{RN}_w(x) := \sum_{k=1}^{n} w_k \ (b_k + v_k x)^+$
- $-b_k/v_k \sim$ g, g probability density w.r.t. Lebesgue measure

Claimed Goal

approximate f_D using standard least squares

$$\min_{w \in \mathbb{R}^n} \frac{1}{N} \sum_{i=1}^{N} (\mathcal{RN}_w(x_i) - y_i)^2 = \min_{w \in \mathbb{R}^n} L(w) \tag{LS}$$

using $\tau := T/\gamma$ iterations of discretized gradient descent
$dw_t = -\nabla L(w_t)\, dt$, $w_0 = 0$ for some stepsize $\gamma > 0$.

RSN and gradient descent - implicit regularization in 1d

Setting

- $D := \{(x_i, y_i) \in \mathbb{R} \times \mathbb{R},\ i = 1, \ldots, N\}$
- $\mathcal{RN}_w : \mathbb{R} \to \mathbb{R}$, $\mathcal{RN}_w(x) := \sum_{k=1}^{n} w_k \ (b_k + v_k x)^+$
- $-b_k/v_k \sim$ g, g probability density w.r.t. Lebesgue measure

Claimed Goal

approximate f_D using standard least squares

$$\min_{w \in \mathbb{R}^n} \frac{1}{N} \sum_{i=1}^{N} (\mathcal{RN}_w(x_i) - y_i)^2 = \min_{w \in \mathbb{R}^n} L(w) \tag{LS}$$

using $\tau := T/\gamma$ iterations of discretized gradient descent
$dw_t = -\nabla L(w_t)\, dt$, $w_0 = 0$ for some stepsize $\gamma > 0$.

RSN and gradient descent - implicit regularization in 1d

compare

1. time T solution $\mathcal{RN}_{w_T}$ of

$$\boxed{\min_{w\in\mathbb{R}^n} \frac{1}{N}\sum_{i=1}^{N}(\mathcal{RN}_w(x_i)-y_i)^2 = \min_{w\in\mathbb{R}^n} L(w)} \qquad \text{(LS)}$$

   using $\tau := T/\gamma$ iterations of discretized gradient descent $dw_t = -\nabla L(w_t)\,dt$, $w_0 = 0$ for some stepsize $\gamma > 0$.

2. smooth spline s_λ, solution to

$$\boxed{\min_{f\in C^2} \frac{1}{N}\sum_{i=1}^{N}(f(x_i)-y_i)^2 + \lambda\int_{\mathbb{R}} \left(f''\right)^2(z)\,dz} \qquad \text{(SP)}$$

   smooth spline interpolation $s := \lim_{\lambda\to 0} \arg\min s_\lambda$

RSN and gradient descent - implicit regularization in 1d

compare

1. time T solution $\mathcal{RN}_{w_T}$ of

$$\boxed{\min_{w\in\mathbb{R}^n} \frac{1}{N}\sum_{i=1}^{N}(\mathcal{RN}_w(x_i)-y_i)^2 = \min_{w\in\mathbb{R}^n} L(w)} \tag{LS}$$

   using $\tau := T/\gamma$ iterations of discretized gradient descent $dw_t = -\nabla L(w_t)\,dt$, $w_0 = 0$ for some stepsize $\gamma > 0$.

2. smooth spline s_λ, solution to

$$\boxed{\min_{f\in C^2} \frac{1}{N}\sum_{i=1}^{N}(f(x_i)-y_i)^2 + \lambda\int_{\mathbb{R}} \left(f''\right)^2(z)\,dz} \tag{SP}$$

   smooth spline interpolation $s := \lim_{\lambda\to 0} \arg\min s_\lambda$

RSN and gradient descent - implicit regularization in 1d

Theorem (Heiss, Teichmann, Wutte (2020))

- $\mathcal{RN}_{w_T(n)} = \sum_{k=1}^{n} w_{T,k}\, \sigma\left(b_k + v_k x\right)$ ... time-T solution of problem (LS)
- s ... smooth spline interpolation

$$\text{Then } \; ||\mathcal{RN}_{w_T} - s||_{W^{1,\infty}(K)} \xrightarrow[T,n\to\infty]{P} 0$$

$$||\mathcal{RN}_{w_T} - s||_{W^{1,\infty}(K)} \leq \underbrace{\left|\left|\mathcal{RN}_{w_T} - \mathcal{RN}_{w^R(n/T)}\right|\right|_{W^{1,\infty}(K)}}_{\xrightarrow[T\to\infty]{} 0}$$

$$+ \underbrace{\left|\left|\mathcal{RN}_{w^R(n/T)} - s\right|\right|_{W^{1,\infty}(K)}}_{\xrightarrow[T,n\to\infty]{P} 0}$$

where the intermediate term is a ridge regression.

RSN and gradient descent - implicit regularization in 1d

Theorem (Heiss, Teichmann, Wutte (2020))

- $\mathcal{RN}_{w_T(n)} = \sum_{k=1}^{n} w_{T,k}\, \sigma\left(b_k + v_k x\right)$ ... time-T solution of problem (LS)
- s ... smooth spline interpolation

$$\text{Then } \; ||\mathcal{RN}_{w_T} - s||_{W^{1,\infty}(K)} \xrightarrow[T,n\to\infty]{P} 0$$

$$||\mathcal{RN}_{w_T} - s||_{W^{1,\infty}(K)} \leq \underbrace{\left|\left|\mathcal{RN}_{w_T} - \mathcal{RN}_{w^R(n/T)}\right|\right|_{W^{1,\infty}(K)}}_{\xrightarrow[T\to\infty]{} 0}$$

$$+ \underbrace{\left|\left|\mathcal{RN}_{w^R(n/T)} - s\right|\right|_{W^{1,\infty}(K)}}_{\xrightarrow[T,n\to\infty]{P} 0}$$

where the intermediate term is a ridge regression.

RSN and gradient descent - implicit regularization in 1d

Theorem (Heiss, Teichmann, Wutte (2020))

- $\mathcal{RN}_{w_T(n)} = \sum_{k=1}^{n} w_{T,k}\, \sigma\left(b_k + v_k x\right)$ ... time-T solution of problem (LS)
- s ... smooth spline interpolation

$$\text{Then } \; ||\mathcal{RN}_{w_T} - s||_{W^{1,\infty}(K)} \xrightarrow[T,n\to\infty]{P} 0$$

$$||\mathcal{RN}_{w_T} - s||_{W^{1,\infty}(K)} \leq \underbrace{\left|\left|\mathcal{RN}_{w_T} - \mathcal{RN}_{w^R(n/T)}\right|\right|_{W^{1,\infty}(K)}}_{\xrightarrow[T\to\infty]{} 0}$$
$$+ \underbrace{\left|\left|\mathcal{RN}_{w^R(n/T)} - s\right|\right|_{W^{1,\infty}(K)}}_{\xrightarrow[T,n\to\infty]{P} 0}$$

where the intermediate term is a ridge regression.

Inverse Problems in Finance: calibration

The calibration problem

Let us first introduce the problem. We have a pool of models Θ, which is parameterized by parameter vector $\theta \in \Theta$, and we have prices of financial products, which can be calculated for a given parameter θ. Take for instance the Black-Scholes model, where the parameter vector $\theta = (S_0, \sigma, r)$ consists of today's price S_0, a volatility parameter σ and an interest rate r. Prices of financial products could be calculated for any path-dependent European or American option.

The calibration problem

Given a such a pool of a models, a price structure which can be calculated given a model $\theta \in \Theta$ and a price structure from real markets, the market price structure (later always referred to as *data*), what is the most appropriate model to choose. *Most appropriate* has to be specified by a loss function

$$\theta \mapsto L^{\mathrm{data}}(\theta)$$

which measures the distance of the model price structure to the market price structure.

Whence we are interested in minimizing $\theta \mapsto L^{\mathrm{data}}(\theta)$ for $\theta \in \Theta$. The calculation of possible minimizers θ^* is called *calibration (problem)*. It is a typical example of an inverse problem, which we have already encountered in the lecture about training. In this setting we do neither develop a theory of regularization, as we have done in the lecure about training, nor do we look into Bayesian approaches, but we rather focus on particular features manifest in calibration problems.

Remarks

1. In typical pricing models (SDE models of affine type, models with easy to evaluate characteristics, etc) it is often relatively easy to evaluate $L^{\mathrm{data}}(\theta)$ for a given parameter $\theta \in \Theta$. Models, where this evaluation is difficult, have been rarely used in industry, for instance dynamic models for implied volatility surfaces or rough volatility models. Still some computational effort lies in this procedure and often only models are used where actually the loss function can be easily evaluated.
2. Even if it is easy to evaluate the loss function given data, the inverse problem is often high dimensional and a high accuracy is desired, i.e. the loss has to be really small. This is in contrast to inverse problems which appear in training where it is often not a priori clear which loss is desirable.
3. Regularization might complicate the problem considerably.

Three ML approaches

- (ambitious approach) learn the map

$$\text{data} \mapsto \theta^*$$

directly. If it works it is of course a wonderful tool, but one needs to be aware that one learns a possibly quite irregular map (if no explicit regularization has been used).

- (modest approach) learn the map

$$\theta \mapsto \text{data}$$

which is relatively easy to generate and often quite regular. Next solve the inverse problem by replacing the pricing functional through its learned approximation.

- (neural model approach) take a modelling approach which includes neural networks right away and train those models to perform respective tasks. In other words enter new modelling paradigms.

Local stochastic volatility models

- In stochastic local volatility (SLV) models the price process $(S_t)_{t\geq 0}$ of one asset follows an SDE
$$dS_t = S_t L(t, S_t)\alpha_t dW_t,$$
where
  - α_t is a stochastic process,
  - $L(t, s)$ is the so called *leverage or local volatiliy function*,
  - W is the driving Brownian motion.
- The function L is the crucial part in this model. It should allow to perfectly calibrate the implied volatility surface seen in the market.
- Given an implied volatility the leverage function has to satisfy
$$L^2(t, s) = \frac{\sigma^2_{\text{Dupire}}(t, x)}{\mathbb{E}[\alpha_t^2 | S_t = s]}.$$
- This is an implicit equation for L since the leverage function is *needed* for the computation of $\mathbb{E}[\alpha_t^2 | S_t = s]$.

Existing methods

The implicit equation for L can be solved via McKean-Vlasov techniques.
The existing calibration techniques can be roughly distinguished into:

- Monte Carlo techniques based on particle methdos for McKean-Vlasov equations:
  - P. Henry-Labordère (2009): "Calibration of Local Stochastic Volatility Models: A Monte-Carlo Approach"
  - J. Guyon and P. Henry-Labordère (2011): "The Smile Calibration Problem Solved"
- PDE techniques based on non-linear Fokker-Planck equations for McKean-Vlasov equations:
  - Y. Tian, Z. Zhu, G. Lee, F. Klebaner, and K. Hamza (2015): "Calibrating and Pricing with a Stochastic-Local Volatility Model".
- inverse problem techniques for PDEs:
  - Y Saporito, X. Yang, J. Zubelli (2017): "The Calibration of Stochastic-Local Volatility Models - An Inverse Problem Perspective"

Deep Calibration

- Choose the parameters of a usual stochastic volatility model.
- Define some grid $0 = t_0 < t_1 \cdots < t_n = T$. Parametrize the leverage function $L(t, s)$ via a family of neural networks, i.e.

$$L(t, s) = F^{i+1}(s) \quad t \in [t_i, t_{i+1}), \quad i \in \{0, \ldots, n-1\},$$

  where for every $i \in \{1, \ldots, n\}$, $F^i \in \mathcal{NN}_{M,1,1}$.
- Parametrize hedging strategies δ (with respect to hedging instruments) by neural networks
- We denote the weights of all the L-networks by θ^L and of the hedging networks by θ^δ.
- Fix θ^L and learn to hedge by changing θ^δ, i.e. minimize a risk functional of **payoff minus market price minus hedge P&L**.
- Fix θ^δ and learn θ^L to price correctly, i.e. minimize the expectation of **payoff minus market price minus hedge**.

Data and Objectives

- Consider several call options on the underlying S parametrized by τ_j corresponding e.g. to different strikes and maturities.
- Fix θ^L and determine the weights θ^δ of all of the hedging networks

$$\text{argmin}_{\theta^\delta} \sum_{j=1}^{J} w_j \mathbb{E}\left[u((S_{T_j} - K_j)_+ - (\delta(\theta^\delta, \tau_j) \bullet S)_{T_j} - \pi^{\text{mkt}}(\tau_j))\right],$$

  where

  - S denote the prices of the underlying depending on θ^L,
  - π^{mkt} denote the market prices of the respective calls,
  - w_j are some product specific weights (e.g. vega weighting).
  - u is a function assessing risk of the position.
- the expectation operator is calculated along N^δ trajectories by MC.

Data and Objectives

- Fix next θ^δ and determine the weights θ^L of all of the L networks

$$\text{argmin}_{\theta^L} \sum_{j=1}^{J} w_j \left(\mathbb{E}\left[(S_{T_j} - K_j)_+ - (\delta(\theta^\delta, \tau_j) \bullet S)_{T_j} - \pi^{\text{mkt}}(\tau_j) \right] \right)^2,$$

- The expectation operator is calculated along N^L trajectories, which can be chosen due to strong variance reducation, extraordinarily small.

Two important remarks

- standard BS hedges are in many cases excellent, hence learning θ^L is often not necessary and does not need to be very precise to do the variance reduction job.
- the variance reduction through hedging is tremendous, i.e. N^L only lies in the thousands.

Example: SABR stochastic local vol model

- In the SABR SLV model the price process $(S_t)_{t\geq 0}$ satisfies

$$dS_t = S_t L(t, S_t)\alpha_t dW_t,$$
$$d\alpha_t = \nu\alpha_t dB_t,$$

  where B and W are correlated Browian motions with $d\langle B_t, W_t\rangle_t = \rho dt$ and ν is the vol of vol.

- where ν, ρ, α_0 are some fixed parameters.
- For the "data" we use some 60 prices at maturities $T = 0.1, 0.25, 0.5, 1.0$ generated by some leverage function.

Implementation

- done in Tensorflow with ADAM optimizer.
- hedging strategies are initialized around the BS hedge.
- goal: re-construct the leverage function (stopping criterion: implied vol error less than 0.001)
- see results at `https://arxiv.org/abs/2005.02505`.

Consistent Recalibration models

There are models of Heath-Jarrow-Morton type where the drift is of complicated nature and only implicitly given, some of those models are consistent recalibration models, see results at `https://arxiv.org/abs/2006.09455`.

References

- C. Cuchiero, W. Khosrawi-Sardroudi, J. Teichmann:
  A generative adversarial network approach to calibration of local stochastic volatility models, arxiv, 2020.
- M. Gambara, J. Teichmann:
  Consistent Recalibration Models and Deep Calibration, submitted, arxiv, 2020.
- J. Heiss, J. Teichmann, H. Wutte;
  How implicit regularization of Neural Networks affects the learned function – Part I, arxiv, 2019.