



Robust Mean Estimation by All Means

Reynald Affeldt  



National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan

Clark Barrett  

Stanford University, CA, USA

Alessandro Bruni  

IT University of Copenhagen, Denmark

Ieva Daukantas  

IT University of Copenhagen, Denmark

Harun Khan  

Stanford University, CA, USA

Takafumi Saikawa  

Nagoya University, Japan

Carsten Schürmann  

IT University of Copenhagen, Denmark

Abstract

We report the results of a verification experiment on an algorithm for robust mean estimation, i.e., an algorithm that computes a mean in the presence of outliers. We formalize the algorithm in the Coq proof assistant and devise a pragmatic approach for identifying and solving issues related to the choice of bounds. To keep our formalization succinct and generic, we recast the original argument using an existing library for finite probabilities that we extend with reusable lemmas. To formalize the original algorithm, which relies on a subtle convergence argument, we observe that by adding suitable termination checks, we can turn it into a well-founded recursion without losing its original properties. We also exploit a tactic for solving real-valued inequalities by approximation to heuristically fix inaccurate constant values in the original proof.

2012 ACM Subject Classification Mathematics of computing → Probabilistic inference problems; Theory of computation → Logic and verification

Keywords and phrases robust statistics, probability theory, formal verification

Digital Object Identifier 10.4230/LIPIcs.ITP.2024.39

Category Short Paper

Supplementary Material

Software (Source code): <https://github.com/affeldt-aist/infotheo/> [1]
archived at [swh:1:dir:0fc3cb212f8a5ba2547161db17ea6b87317bad39](https://swh.1:dir:0fc3cb212f8a5ba2547161db17ea6b87317bad39)

Funding The first and sixth authors acknowledge support of the JSPS KAKENHI Grant Number 22H00520. The second and fifth authors were supported in part by the Stanford Center for Automated Reasoning.

Acknowledgements The authors would like to thank the anonymous referees for their thorough reviews.

1 Towards formally-verified robust mean estimators

Our motivation is to produce formally-verified programs that perform robust mean estimation as a first step towards formal robust statistics. The setting for robust mean estimation is as follows. We are given samples from an unknown distribution, but some fraction of them



© Reynald Affeldt, Clark Barrett, Alessandro Bruni, Ieva Daukantas, Harun Khan, Takafumi Saikawa, and Carsten Schürmann;

licensed under Creative Commons License CC-BY 4.0

15th International Conference on Interactive Theorem Proving (ITP 2024).

Editors: Yves Bertot, Temur Kutsia, and Michael Norrish; Article No. 39; pp. 39:1–39:8



Leibniz International Proceedings in Informatics

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

are *outliers* (data points that differ significantly from other observations) that we want to discard. A robust mean estimator is an algorithm that computes the mean of a set of data points while minimizing the effect of outliers. More formally, we say that a mean estimator is *robust* when the difference between the computed mean and the optimal mean, without considering outliers, can be upper-bounded by arbitrarily small positive numbers. There are several algorithms for robust mean estimation, e.g., the median is a robust estimator, and the trimmed mean by Tukey [14] is a robust estimator. Both of these estimators work by excluding samples. The robustness of trimmed mean has already been formalized in COQ [5].

In this paper we verify the robustness of an archetypal algorithm for mean estimation that operates by iteratively re-weighting the influence of each sample. The algorithm is an example of an M-estimator [7, Ch. 3], and it is described in Steinhardt’s Ph.D. thesis [11] under the name `filter1d`. M-estimators assign weights to each sample instead of excluding samples. Not all M-estimators are robust: robustness requires that sufficiently low weights are assigned to outliers.

We identify and resolve two main challenges with the algorithm `filter1d`. First, through formalization, we identify and fix issues with the original proofs which contain erroneous constants and bounds. The second challenge is technical. The original proofs spelled out calculations in terms of “big sums,” which are the intuitive *lingua franca* for probability theory in finite settings. We use and extend existing libraries to keep the formalization succinct and reusable, and as a consequence, proofs became more modular.

These challenges, checking and occasionally fixing constant bounds, and devising techniques for symbolic calculations, are recurring issues when formalizing paper proofs about statistical algorithms. In this paper, we formalize the `filter1d` algorithm and its related theory in COQ [12]. We leverage this experience to inform a broader discussion of these recurring issues and to document a general approach. We represent calculations involving probabilities using an existing library, and we also take advantage of an automated tactic in COQ (namely `interval` [9]) to fix erroneous constants in the original proof. In the end, we are able to formalize `filter1d` without ambiguity in the language of the COQ proof assistant.

The paper is organized as follows. First, we present the `filter1d` algorithm for robust mean estimation in Sect. 2. In Sect. 3, we explain how we formalize the bound on the mean using an existing library and using changes of distributional assumptions. In Sect. 4, we explain how we formalize the bound on the variance by fixing the original proofs, in practice by using the `interval` tactic of COQ. Finally, we formalize the algorithm for robust mean estimation and prove its termination and correctness in Sect. 5. The results presented in this paper are available online as a COQ formalization [1].

2 An archetypal algorithm for robust mean estimation

The algorithm takes as input a discrete random variable X (with sampled values x_1, \dots, x_n), a finite probability distribution P (with probabilities p_1, \dots, p_n), and a positive value v . It either computes a mean $\hat{\mu}$ or fails. The computed mean is expected to be close to the mean of the random variable X for the distribution P with the outliers removed. It is correctly computed in the following sense: if there exists a subset S of $\text{dom}(X)$ such that $v = \sigma_S^2$ and the probability ε of \bar{S} (i.e., the set of outliers) is smaller than $1/12.7$, then $|\hat{\mu} - \mu_S|$ is bounded by $\sqrt{\frac{v \cdot 2\varepsilon}{2 - \varepsilon}} + \sqrt{\frac{16v \cdot 2\varepsilon}{1 - \varepsilon}}$ (where $\sigma_S^2 \stackrel{\text{def}}{=} \mathbb{V}[X|S]$ and $\mu_S \stackrel{\text{def}}{=} \mathbb{E}[X|S]$). This signifies what is meant by robustness: as long as the probability of the set of outliers is smaller than a constant, the result is guaranteed to stay within a reasonable range.

The algorithm maintains a sequence of weights c_i , initialized to 1, which represent the contribution of each point to the computation. These weights are then updated iteratively, such that points that deviate the most are given less weight (Algorithm 1).

■ **Algorithm 1** `filter1d`.

1. Initialize each weight c_1, \dots, c_n to 1
 2. Compute the *empirical mean*: $\hat{\mu}_c \leftarrow (\sum_{i=1}^n p_i x_i c_i) / (\sum_{i=1}^n p_i c_i)$
 3. Compute the *empirical variance*: $\hat{\sigma}_c^2 \leftarrow (\sum_{i=1}^n p_i c_i \tau_i) / (\sum_{i=1}^n p_i c_i)$ where $\tau_i \stackrel{\text{def}}{=} (x_i - \hat{\mu}_c)^2$
 4. If $\hat{\sigma}_c^2 \leq 16v$ then terminate and output $\hat{\mu}_c$
 5. Otherwise, update $c_i \leftarrow c_i(1 - \tau_i/\tau_{max})$ where $\tau_{max} = \max_{i \in \text{supp}(c)} \tau_i$
 6. If all $c_i = 0$ then terminate with error; otherwise, go back to line 2
-

Note that at Step 5, we take the maximum over the support of the function $c : i \mapsto c_i$ whereas the original algorithm [11, Sect. 1.2.3] does not make this explicit. This makes our algorithm slightly different from the original algorithm, but it is a reasonable modification because it does not change the computed values. In Step 6, our algorithm also checks that not all c_i are zero before continuing, because otherwise there is a division by zero in Step 2 (take for example the situation of a positive computed variance and two points of equal weight). These modifications do not change the property originally stated by Steinhardt, namely that when `filter1d` terminates, it results in the desired mean (Step 4). Furthermore, we generalize the algorithm by giving each point x_i a probability p_i instead of assuming a uniform distribution¹ as in [11].

Lastly, the robustness of `filter1d` is a consequence of the following invariant [11, eqn (\mathcal{I}), page 5] being preserved: $\sum_{i \in S} (1 - c_i) p_i \leq \frac{1-\epsilon}{2} \sum_{i \in \bar{S}} (1 - c_i) p_i$ (\mathcal{I}). It shows that the amount of “mass” (the sum of the weights) removed from the points in S is less than $\frac{1-\epsilon}{2}$ times the amount of mass removed from the outliers. The invariant is key to establishing the bound between the empirical mean and the mean on the points in S shown in Sect. 3.3, which delivers the final robustness argument, and the bounding of the empirical variance of Sect. 4, which in turn is necessary to show that the invariant is preserved when the weights are updated.

3 Bounding the empirical mean using resilience and changes of distributional assumptions

In this section, we rework the original proofs so that they can be formalized using INFO_{THEO} [3, 4], a formalization of information theory in COQ.

3.1 Background about formalization of probabilities

INFO_{THEO} introduces a number of definitions and lemmas to deal with finite probabilities. The type of finite distributions is `{fdist U}` where U is a finite type (`finType` as provided by the MATHCOMP library [8]). A finite set in MATHCOMP is an object of type `{set U}` where

¹ This change does not affect the final computation: for finite samples, p_i corresponds to a multiplicity count of the occurrences of x_i . In the case of a uniform distribution, all p_i have the value $1/n$, so the original algorithm is a special case. Note that the computation of the updated c_i at Step 5 does not directly depend on the value of p_i .

39:4 Robust Mean Estimation by All Means

U is a finite type; finite sets are used to represent events. A probability space is a finite type with a function P , which assigns to each point of the type a probability in $[0, 1]$. The domain of P can also be extended to finite sets, taking the sum of the pointwise values, resulting in the corresponding probability measure. A random variable is a function embodying the notion of probabilistically changing values: it goes from a probability space to a type, the input being assumed to be sampled from the probability space according to its probability measure. The type of real-valued random variables is denoted by $\{\text{RV } P \rightarrow \mathbb{R}\}$ where P is the probability measure of the ambient probability space and \mathbb{R} is the type of real numbers [3, Sect. 2]. The conditional expectation of the random variable X w.r.t. an event A is $\mathbb{E}[X \mid A]$ (see [2] for conditional probabilities in `INFOTHEO`). We use a *resilience* lemma that bounds the distance between two conditional expectations (generalizing [5, Sect. 5.1]):

► **Lemma 1 (Resilience).** *Let X be a random variable with probability measure P , and F, G be two events such that $F \subseteq G$. Then for any δ such that $0 < \delta \leq P(F)/P(G)$, we have*

$$|\mathbb{E}[X \mid F] - \mathbb{E}[X \mid G]| \leq \sqrt{2\text{V}[X \mid G] \frac{1 - \delta}{\delta}}.$$

3.2 Changing distributional assumptions

Steinhardt argues for the correctness of his algorithm using big sums. We recast big sums in terms of expectation and variance, as provided by `INFOTHEO`, to enable the reuse of existing lemmas. We further extend `INFOTHEO` with lemmas about changes in distributional assumptions of RVs.

A *change of distributional assumption* is the transformation of a RV X on a probability space $T1$ with probability measure P into another RV on another space $T2$ with probability measure Q by precomposing a function $f : T1 \rightarrow T2$:

Definition `change_dist` ($T1\ T2 : \text{finType}$) ($P : \{\text{fdist } T1\}$) ($Q : \{\text{fdist } T2\}$)
 $(f : T2 \rightarrow T1)$ ($X : \{\text{RV } P \rightarrow \mathbb{R}\}$) : $\{\text{RV } Q \rightarrow \mathbb{R}\} := X \ \backslash \circ \ f$.

We denote with $Q.\text{-RV } X \ \backslash \circ \ f$ the resulting RV and write $Q.\text{-RV } X$ when f is the identity.

The main application of changing distributional assumptions is to formalize the empirical variance (Step 3 of `filter1d`). Given a probability measure P and (non-negative) weights c_i ($i \in A$), we call the probability measure $i \mapsto c_i p_i / \sum_{j \in A} c_j p_j$ *weighted*. In `COQ` we provide the weighted probability measure of P as a function `wgt` whose argument is a proof of $\sum_{j \in A} c_j p_j \neq 0$. We call *weighted* the change of distributional assumption with a weighted probability measure. In particular, the *empirical mean* of X with weights c_i can be expressed as the expectation of a weighted RV.

Similarly, given a probability measure P over U and a function h with codomain $\subseteq [0, 1]$, we call *split* the probability measure over $U \times \{F, T\}$ (`Split.d P h` in `COQ`):

$$\text{split}(P, h)_{(i,b)} \stackrel{\text{def}}{=} \begin{cases} h(i)p_i & \text{if } b = T \\ (1 - h(i))p_i & \text{if } b = F. \end{cases}$$

We call *first-split* the RV $Q.\text{-RV } X \ \backslash \circ \ \text{fst}$ (where Q is `Split.d P h`). It is possible to change a conditional expectation by a first-split. (The `COQ` notation $\backslash *$ is for the Cartesian product and `[set: bool]` is for the set of booleans.)

Definition `fst_RV` ($X : \{\text{RV } P \rightarrow \mathbb{R}\}$) : $\{\text{RV } d \rightarrow \mathbb{R}\} := (\text{Split.d } P \ h).\text{-RV } X \ \backslash \circ \ \text{fst}$.
Lemma `cEx` ($X : \{\text{RV } P \rightarrow \mathbb{R}\}$) $A : \mathbb{E}[X \mid A] = \mathbb{E}[\text{fst_RV } X \mid A \ \backslash * \ [\text{set: bool}]]$.

3.3 Bounding the empirical mean

We prove the following bound between the mean and the empirical mean under the invariant (\mathcal{I}) : $|\mu_S - \hat{\mu}_c| \leq \sigma_S \sqrt{\frac{2\varepsilon}{2-\varepsilon}} + \hat{\sigma}_c \sqrt{\frac{2\varepsilon}{1-\varepsilon}}$ [11, Lemma 1.4].

For that purpose, we introduce an intermediate value $\tilde{\mu}_c$ and first show $(\mu_S - \tilde{\mu}_c)^2 \leq \hat{\sigma}_c^2 \frac{2\varepsilon}{1-\varepsilon}$, which is formalized as follows, using PCO, a proof that `Weighted.total P C != 0`:

`Let WP := Weighted.d PCO.`

`Lemma bound_emean : invariantW -> (* A weaker version of the invariant *)
 (E_[WP.-RV X | S] - E_(WP.-RV X))^+2 <= V_(WP.-RV X) * 2 * eps / (1 - eps).`

The proof is a direct application of Lemma 1 with $\delta = 1 - \varepsilon$ and G being the full set.

The second bound is $(\mu_S - \tilde{\mu}_c)^2 \leq \sigma_S^2 \frac{2\varepsilon}{2-\varepsilon}$, proved using the inequality $1 - \varepsilon/2 \leq \frac{\sum_{i \in S} c_i p_i}{P(S)}$ (which we shall call ‘‘S-mass’’):

`Lemma bound_mean : invariant ->
 (E_[X | S] - E_[WP.-RV X | S])^+2 <= V_[X | S] * 2 * eps / (2 - eps).`

The proof relies on the observation that one can change the distributional assumption of μ_S as $\mathbb{E}[X_1|S \times \{F, T\}]$ and similarly $\tilde{\mu}_c = \mathbb{E}[X_1|S \times \{T\}]$, where X_1 is the first-split of X . This changing of distributional assumption corresponds to the formalization of the following proof step in [11, page 63]: ‘‘(here we think of $\tilde{\mu}_c$ as the mean of an event occurring with probability $\sum_{i \in S} c_i/|S|$ under the uniform distribution on $|S|$)’’. Using changing of distributional assumption, the proof is actually an application of Lemma 1, using ‘‘S-mass’’ to fulfill its hypothesis:

$$\underbrace{(\mathbb{E}[X_1|S \times \{F, T\}] - \mathbb{E}[X_1|S \times \{T\}])^2}_{\mu} \leq \underbrace{2 \mathbb{V}[X_1|S \times \{F, T\}]}_{\mathbb{V}[X|S]} \frac{1 - (1 - \varepsilon/2)}{1 - \varepsilon/2}.$$

↑
Lemma 1

4 Bounding of variance: using interval to fix proofs

In this section, we explain how we formalize and fix the proofs that bound the empirical variance of `filter1d`. Precisely, we obtain the following bound for the empirical variance.

► **Lemma 2.** *Provided the invariant (\mathcal{I}) , $16\sigma_S^2 \leq \hat{\sigma}_c^2$, and $\varepsilon \leq 1/12.7$, we have:*

(a) $\sum_{i \in S} c_i p_i \tau_i \leq \frac{1-\varepsilon}{3.35} \sigma_c^2$ and (b) $\sum_{i \in \bar{S}} c_i p_i \tau_i \geq \frac{2}{3.35} \sigma_c^2$.

Steinhardt claims an upper-bound of $1/12$ for ε and uses 3 in the denominators in (a) and (b) instead of 3.35 [11, Lemma 1.4 (part 2)/eqn A.6–A.9]. We believe that the lemma cannot be proved with Steinhardt’s bounds because we corrected mistakes in the original proof of (b) [11, eqn A.10–A.11, page 63], which we discovered when we mechanized an argument that ‘‘consist[s] of straightforward but tedious calculation’’ [11, page 5].

The correct bounds can be obtained by using the `interval` tactic [9, 10] of Coq. We parameterize the Coq statements corresponding to Lemma 2 with a variable `eps_max` for the upper-bound to be found and with a variable `denom` for the denominators in (a) and (b):

`Notation eps := Pr P cplt_S. (* cplt_S is the complement of S *)
 Notation eps_max := 10 / 127. (* values found by try-and-error, see below *)
 Notation denom := 335 / 100.
 Hypothesis low_eps : eps <= eps_max.`

`Lemma bound_empirical_variance_S :
 \sum_(i in S) C i * P i * tau i <= (1 - eps)/denom * V_(WP.-RV X).`

`Lemma bound_empirical_variance_cplt_S :
 2/denom * V_(WP.-RV X) <= \sum_(i in cplt_S) C i * P i * tau i.`

Then, we use two proof scripts (inspired by Steinhard’s proofs) to reduce the proof to purely arithmetical subgoals depending on the variables `eps_max` and `denom`. Finally, we use COQ to help find optimal values for `eps_max` and `denom` by adjusting the parameters and replaying the proof scripts, deferring subgoals related to arithmetic to `interval`. In this way, we obtained the values 3.35 and 1/12.7 for `denom` and `eps_max`, respectively, by iterative trial-and-error. Note that with Steinhard’s parameters (3 and 1/12), the proof script `bound_empirical_variance_S` holds, but `bound_empirical_variance_cplt_S` does not. It is possible to use the same process to further refine the constants: e.g., we are able to show the same results for 3.345 and 1/12.65.

5 A formally robust algorithm for mean estimation

5.1 Formalizing `filter1d`

The algorithm `filter1d` was presented in Sect. 2 in the form of a loop. To formalize it in COQ, we turn it into a recursive algorithm by using the `Function` command [12, Functional Induction], which can be used for arbitrary well-founded recursion (not just structural).

```

1 Variables (U : finType) (P : {fdist U}) (X : {RV P -> R}).
2 Function filter1D_rec v (v_ge0 : 0 <= v)
3   (C : nneg_finfun U) (C01 : is01 C) (PC0 : Weighted.total P C != 0)
4   {measure (fun C => #| 0.-support C |) C} :=
5   let WP := wgt PC0 in
6   if `V (WP.-RV X) <=? 16 * v is left _ then
7     Some (`E (WP.-RV X))
8   else
9     let C' := update X PC0 in
10    if Weighted.total P C' !=? 0 is left PC0' then
11      filter1D_rec v_ge0 (is01_update X PC0 C01) PC0'
12    else
13      None.
```

The parameter `U` is a finite type, `P` is a probability measure over `U`, and `X` is a random variable whose probability measure is `P`. The function takes as parameters the variance `v` with a proof that it is non-negative (`v_ge0`), as well as a (non-negative) weight function `C` with proofs that the weights are less than 1 (`C01`) and that their total is not 0 (`PC0`). The measure that controls termination is the size of the support of `C`. Indeed, an execution of Step 5 sets one nonzero weight to zero (the weight C_i such that $\tau_i = \tau_{max}$), rendering the nonzero support of C_i ’s (`0.-support C` at line 4) strictly smaller.

At each iteration, we update the distributional assumption (line 5) and compute the variance (line 6). Then, we update the weights (line 9) and test if we can recurse by checking that the total of the new weights is nonzero. A termination is reached if (1) the empirical variance satisfies the convergence condition $\hat{\sigma}_c^2 \leq 16v$ (line 6), resulting in `Some($\hat{\mu}_c$)`, or (2) the weighted total after the update is zero, resulting in `None` (line 13). When neither is the case, we perform a recursive call at line 11, passing two proof terms that also implicitly carry the updated C'_i ’s. The function `is01_update` computes a term that shows that C'_i ’s do not break the invariant that they are between 0 and 1. The proof `PC0'` directly results from the case analysis at line 10.

We feed a set of constant weights, all equal to 1, as the base case of C_i to complete the definition of `filter1d` (again, in this definition, the constant weights are implicitly passed by two proof terms `C1_is01 U` and `PC1_neq0 P`):

Definition `filter1D v (v_ge0 : 0 <= v) := filter1D_rec v_ge0 (C1_is01 U) (PC1_neq0 P)`.

5.2 Robustness of `filter1d`

We can finally prove the robustness of `filter1d`. The recursive definition based on the measure `#|0.-support C|` ensures that the function returns `None` if and only if the computation has failed and all C_i 's are set to zero. Otherwise, it returns the empirical mean upon termination.

The first step in proving robustness is to show the preservation of the invariant (\mathcal{I}) w.r.t. the update performed at Step 5:

```
Lemma invariant_update : let C' := update X PCO in
  invariant P C S eps -> invariant P C' S eps.
```

This formalizes [11, Lemma 1.5, page 5]. The proof is a consequence of Lemma 2 (Sect. 4) used in conjunction with the following property of `update` (`update_removed_weight` in [1]): $\sum_{i \in E} p_i(1 - c'_i) = \sum_{i \in E} p_i(1 - c_i) + 1/\tau_{max} \sum_{i \in E} p_i c_i \tau_i$ where the c'_i 's are the weights after an update, i.e., $c'_i := c_i \left(1 - \frac{\tau_i}{\tau_{max}}\right)$.

Using the preservation of the invariant, we show that `filter1d` is robust with the following theorem and its corollary:

```
1 Hypothesis low_eps : eps <= eps_max.
2 Lemma filter1D_correct :
3   let v := `V_[X | S] in
4   if @filter1D U P X v v_ge0 is Some mu_hat
5   then `| `E_[X | S] - mu_hat | <= Num.sqrt (v * (2 * eps) / (2 - eps)) +
6     Num.sqrt (16 * v * (2 * eps) / (1 - eps))
7   else false.
8
9 Corollary filter1D_converges : @filter1D U P X `V_[X | S] v_ge0 != None.
```

The `@` mark (in lines 4 and 9) disables the inference of implicit arguments in COQ. The theorem shows that, if given the appropriate ε and variance: (i) the algorithm never terminates with an error; and (ii) when the algorithm terminates, the empirical mean is close enough to the true mean.

6 Conclusions

This paper describes a mechanized proof of a simple M-estimator in the proof assistant COQ. The result contributes to the state of the art by improving the pencil-and-paper presentation, by making explicit all details, by clarifying the termination argument, by fixing errors in the pencil-and-paper proofs, and by improving the error bounds. Our approach makes use of the `interval` tactic, which helped determine the correct bounds in the formulation of the main lemma (Sect. 4). The formal verification of the M-estimator can be seen as yet another result in a series of applications of formalized probability, among them the verification of stochastic algorithms [13] and machine learning algorithms [15].

In future work, we plan to generalize the present work on robust mean estimation to the multi-dimensional case (as in [6]) and to normed vector spaces.

References

- 1 Reynald Affeldt, Alessandro Bruni, Ieva Daukantas, and Takafumi Saikawa. Robust mean estimators. Available as part of the InfoTheo library [4], directory `robust`, 2024.
- 2 Reynald Affeldt, Jacques Garrigue, and Takafumi Saikawa. Reasoning with conditional probabilities and joint distributions in Coq. *Computer Software*, 37(3):79–95, 2020. doi: 10.11309/jssst.37.3_79.

- 3 Reynald Affeldt, Manabu Hagiwara, and Jonas Sénizergues. Formalization of Shannon’s theorems. *J. Autom. Reason.*, 53(1):63–103, 2014. doi:10.1007/S10817-013-9298-1.
- 4 Reynald Affeldt, Manabu Hagiwara, Jonas Sénizergues, Jacques Garrigue, Kazuhiko Sakaguchi, Taku Asai, Takafumi Saikawa, Naruomi Obata, and Alessandro Bruni. InfoTheo: A Coq formalization of information theory and linear error-correcting codes. <https://github.com/affeldt-aist/infotheo>, 2018. Last stable release: 0.7.2 (2024).
- 5 Ieva Daukantas, Alessandro Bruni, and Carsten Schürmann. Trimming data sets: a verified algorithm for robust mean estimation. In *23rd International Symposium on Principles and Practice of Declarative Programming (PPDP 2021), Tallinn, Estonia, September 6–8, 2021*, pages 17:1–17:9. ACM, 2021. doi:10.1145/3479394.3479412.
- 6 Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high-dimensions without the computational intractability. *SIAM J. Comput.*, 48(2):742–864, 2019. doi:10.1137/17M1126680.
- 7 Peter J. Huber. *Robust Estimation of a Location Parameter*, pages 492–518. Springer New York, New York, NY, 1992. doi:10.1007/978-1-4612-4380-9_35.
- 8 MathComp. The mathematical components library. Available at <https://github.com/math-comp/math-comp>, 2007. Last stable version: Version 2.2.0 (2024).
- 9 Guillaume Melquiond. <https://coqinterval.gitlabpages.inria.fr/>, 2008. Last stable version: 4.12.0 (2024).
- 10 Guillaume Melquiond. Proving bounds on real-valued functions with computations. In *4th International Joint Conference on Automated Reasoning (IJCAR 2008), Sydney, Australia, August 12–15, 2008*, volume 5195 of *Lecture Notes in Computer Science*, pages 2–17. Springer, 2008. doi:10.1007/978-3-540-71070-7_2.
- 11 Jacob Steinhardt. *Robust Learning: Information Theory and Algorithms*. PhD thesis, Stanford, 2018.
- 12 The Coq Development Team. *The Coq Proof Assistant Reference Manual*. Inria, 2024. Available at <https://coq.inria.fr>. Version 8.19.0.
- 13 Jean-Baptiste Tristan, Joseph Tassarotti, Koundinya Vajjha, Michael L. Wick, and Anindya Banerjee. Verification of ML systems via reparameterization, 2020. arXiv:2007.06776.
- 14 J.W. Tukey and Princeton University. Department of Statistics. *A Survey of Sampling from Contaminated Distributions*. STRG Technical report. Princeton University, 1959.
- 15 Koundinya Vajjha, Barry M. Trager, Avraham Shinnar, and Vasily Pestun. Formalization of a stochastic approximation theorem. In *13th International Conference on Interactive Theorem Proving (ITP 2022), August 7–10, 2022, Haifa, Israel*, volume 237 of *LIPICs*, pages 31:1–31:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.ITP.2022.31.