KESSLER:
A MACHINE LEARNING LIBRARY FOR SPACECRAFT COLLISION AVOIDANCE

Giacomo Acciarini[1], Francesco Pinto[2], Francesca Letizia[3], José A. Martinez-Heras[4], Klaus Merz[5], Christopher Bridges[6], and Atılım Güneş Baydin[7]

[1]*Aerospace Center of Excellence, University of Strathclyde, 75 Montrose Street, G1 1XJ, Glasgow, United Kingdom, Email: giacomo.acciarini@gmail.com*

[2]*Dept. of Engineering Science, University of Oxford, Parks Road, OX1 3PJ, Oxford, United Kingdom, Email: francesco.pinto@eng.ox.ac.uk*

[3]*IMS Space Consultancy at ESA/ESOC, Robert-Bosch-Str. 5, 64293 Darmstadt, Germany, Email: francesca.letizia@esa.int*

[4]*Solenix GmbH, Spreestrasse 3, 64295, Darmstadt, Germany, Email: jose.antonio.martinez.heras@esa.int*

[5]*Space Debris Office, European Space Agency, Robert-Bosch-Str. 5, 64293 Darmstadt, Germany, Email: klaus.merz@esa.int*

[6]*Surrey Space Centre, BA building, University of Surrey, GU2 7XH, Guildford, Surrey, United Kingdom, Email: c.p.bridges@surrey.ac.uk*

[7]*Dept. of Engineering Science & Dept. of Computer Science, University of Oxford, Parks Road, OX1 3PJ, Oxford, United Kingdom, Email: gunes@robots.ox.ac.uk*

ABSTRACT

As megaconstellations are launched and the space sector grows, space debris pollution is posing an increasing threat to operational spacecraft. Low Earth orbit is a junkyard of dead satellites, rocket bodies, shrapnels, and other debris that travel at very high speed in an uncontrolled manner. Collisions at orbital speeds can generate fragments and potentially trigger a cascade of more collisions endangering the whole population, a scenario known since the late 1970s as the Kessler syndrome. In this work we present Kessler: an open-source Python package for machine learning (ML) applied to collision avoidance. Kessler provides functionalities to import and export conjunction data messages (CDMs) in their standard format and predict the evolution of conjunction events based on explainable ML models. In Kessler we provide Bayesian recurrent neural networks that can be trained with existing collections of CDM data and then deployed in order to predict the contents of future CDMs in a given conjunction event, conditioned on all CDMs received up to now, with associated uncertainty estimates about all predictions. Furthermore Kessler includes a novel generative model of conjunction events and CDM sequences implemented using probabilistic programming, simulating the CDM generation process of the Combined Space Operations Center (CSpOC). The model allows Bayesian inference and also the generation of large datasets of realistic synthetic CDMs that we believe will be pivotal to enable further ML approaches given the sensitive nature and public unavailability of real CDM data.

Keywords: spacecraft collision avoidance, space debris, machine learning, probabilistic programming

1. INTRODUCTION

With the advent of the New Space era and the launch of megaconstellations the population of human-made objects in orbit is constantly growing, together with the risk of collisions among objects. Currently, more than 30,000 objects are known to be orbiting the Earth, including a small percentage of payloads of active missions, and a vast population of rocket bodies, mission related objects, debris, and inactive satellites. Several studies exist that analyze the future growth of space objects and warn about the associated increase in collision risks [30, 29, 38, 34].

In 1978, Donald J. Kessler and Burton Cour-Palais, scientists at the NASA Johnson Space Center, conjectured a theoretical scenario where where a few collisions could produce orbiting fragments that would in turn increase the probability of further collisions and potentially trigger a chain reaction leading to the growth of a debris belt around the Earth [22], a process that has been known as the Kessler syndrome since the 1980s. As past collision events demonstrated [4], a collision among satellites can release thousands of debris pieces, which travel at extremely high velocities, therefore representing a big threat for the neighboring population of resident space objects.

For these reasons, international institutions and space agencies constantly release and update international guidelines for the mitigation of collision risk and the safeguarding of the space environment [9]. Moreover, satellite operators continuously monitor operational satellites and perform collision avoidance actions (e.g., maneuvers) for reducing the collision risk. The US Strategic Command (USSTRATCOM) continuously tracks resident space ob-

jects via a global Space Surveillance Network (SSN) and updates a public catalog of two-line element (TLE) data, where each epoch some environment model parameters are stored together with orbital element information from which the current position and velocity of resident space objects can be estimated.

Meanwhile, the Combined Space Operations Center (CSpOC) propagates the SSN observations several days into the future, and monitors possible collision events with other objects in space. If a conjunction has been projected according to some screening criteria, then a "Conjunction Data Message" (CDM) [12] is released to the owner/operator of the satellite. This message contains information about the conjunction event at the predicted time of closest approach (TCA), such as environment aspects used for the orbit propagation, as well as the predicted states of the screened pair of objects, and their associated uncertainties (in the form of covariances). Furthermore, as more observations are processed by CSpOC, new CDMs are released in the week leading up to TCA. Usually, operators are provided with a time-series of CDMs that they use for analyzing each conjunction event and for planning risk mitigation strategies when needed [8, 33, 11].

The tasks of assessing the collision risk of a conjunction event and devising an optimal strategy to mitigate collision risk place a significant burden on space operators that is expected to increase as more objects are launched in space and the space sector grows [28]. Several ways of tackling the space debris problem have been studied by companies, space agencies and researchers in the last years [25, 46, 31, 24]. Recently, the European Space Agency (ESA) has started an international competition called the Collision Avoidance Challenge[1] to study the possibility of helping human operators through machine learning (ML) based approaches that can model and predict conjunction events, therefore alleviating the burden on operators [42]. During the competition, a dataset of thousands of CDMs collected by ESA from 2015 to 2019 has been released, which we call the Kelvins dataset [41]. Although this represents the first public release of CDMs, the data were partially anonymized, hiding the full state (position and velocity) of the satellites, as well as any absolute time information (e.g., TCA, CDM creation time) and other information (e.g., objects' identities).

We believe that ML approaches will be essential in improving collision avoidance analyses and decision making processes in the near future and enable scalable automated systems that would greatly enhance the collision avoidance process, with positive consequences for the safeguarding of the space environment. Following up on the ESA studies, we develop Kessler (Figure 1) a ML library for spacecraft collision avoidance, that we named in honor of the NASA scientist Donald J. Kessler known for proposing the Kessler syndrome. To deliver maximum benefit to the space and ML communities, we release Kessler as an open-source project that we expect to keep improving following this initial release and to which we welcome contributions from the wider community.

Figure 1. Kessler is an open-source library released at `https://github.com/kesslerlib/kessler`

Kessler is an open-source Python package that currently includes Bayesian ML and probabilistic programming components. The library currently provides the following key capabilities:

- Functionality to import and export CDM data, using either the CDM standard format or databases that can be connected to Kessler through an API based on pandas[2] `DataFrame` objects, and grouping CDMs into `Event` objects representing conjunctions.
- Plotting code to visualize event evolution either in a sequence of existing CDMs in a conjunction event, or with future CDMs predicted by the ML modules, representing event evolution predictions with associated uncertainty information.
- A ML module that currently implements Bayesian recurrent neural networks, specifically based on a stack of the long short-term memory (LSTM) architecture [18] that we designed and evaluated to work well in this setting [37], ready to train with the user's own collection of CDM data.
- A probabilistic programming module simulating conjunction events and CDM generation processes, which can be used for either performing event analysis using Bayesian inference or generation of synthetic CDM datasets sampled from this probabilistic generative model.

In the following, we will describe the package and its key features. In particular, we will first discuss related work and methodology on which the main Kessler tools are based on. Then we will detail the software architecture, show some of the package functionalities, and present our conclusions and recommendations for future work.

[1] `https://kelvins.esa.int/collision-avoidance-challenge/`

[2] `https://pandas.pydata.org/`

2. RELATED WORK

2.1. Spacecraft collision avoidance

As previously discussed, operators regularly receive CDM updates describing target and chaser objects in a conjunction event and their associated uncertainties, in the week leading up to TCA. Optionally, the data is augmented with other sources (e.g., other radar measurements, telemetry). Given this collection of information, operators usually perform two steps. First, in case more data are available, they combine the information and assess the state of the two objects and their uncertainties at TCA, for which several techniques exist and are used. These range from linear/linearised methods such as Kalman filters [43], to semi-analytic techniques such as differential algebra with Gaussian mixture models, and polynomial chaos expansion [39, 20, 45, 2], or sampling-based techniques such as Monte Carlo [13]. Second, they process the CDM information and their own analysis (if relevant) to obtain collision risk estimates for the studied conjunction event. Usually the more the time approaches TCA, the more CDMs contain precise (i.e., less uncertain) information. Therefore, in a typical scenario, the operators continuously update these risk estimates, until one or two days before TCA, where they shall make a decision on whether to maneuver the craft or not in order to avoid a collision.

2.2. Bayesian machine learning

Although ML, mainly in the form of deep learning [15, 27], has been extremely successful in providing state-of-the-art results in many complex applications, it is well known that the predictions of neural networks can be over-confident even when the network produces a wrong prediction [3, 26, 40]. This problem is one of the main factors to consider in the application of neural networks to safety-critical applications. To address this issue, ML research has delivered several techniques to augment neural networks with reliable uncertainty measures that should clearly indicate when the model is confident in its predictions or not. Some of these techniques are ascribed to the family of Bayesian deep learning models [47, 21, 7], where the idea is to design models that can produce probability distributions as outputs (as opposed to single point estimates), and to measure the uncertainty on a prediction as a spread over such distributions (e.g., variance or entropy). This can be achieved by replacing the weights of a neural network with probability distributions over these weights, learned via an inference procedure at training time. Since applying inference is usually computationally expensive, many approximate inference schemes have been designed. One of the most popular approximate inference schemes is Monte Carlo dropout, where the idea is to apply dropout, originally proposed as a regularization scheme for neural network training [17], both at training and test time [14]. Dropout is a technique that randomly switches off a proportion of the units of a neural network by sampling a binary mask according to a Bernoulli distribution. It has been shown that, if applied at test time, this technique can be considered a variational approximate inference scheme. For this reason, the distribution produced by feeding the same input to the neural network multiple times with dropout (i.e., resampling the mask that switches off different units in each evaluation) can be used to quantify the uncertainty of the trained neural network.

2.3. Probabilistic programming

Probabilistic programming [16, 44] is a paradigm for statistical modeling that enables automated Bayesian inference in probabilistic models implemented using general-purpose programming languages. It is a technique that can be classified under the larger field of simulation-based inference [10]. Recently techniques have been developed to enable the use of existing stochastic simulator code bases as probabilistic programs so that Bayesian inference in these simulators can be performed [5, 6], creating a new connection between a very large collection of accurate large-scale simulators in many application domains and probabilistic inference techniques developed by the ML community. The idea in probabilistic programming is to leverage domain knowledge to design a simulator (a model) that accurately reproduces the phenomenon of interest. The simulator includes a collection of latent random variables $\boldsymbol{x}$, each of which is assumed to be distributed according to a prior distribution in a Bayesian setting, jointly referred to as $p(\boldsymbol{x})$, expressing the domain knowledge. The joint prior distribution encodes the expert's knowledge about the values that $\boldsymbol{x}$ can assume in the given application and the belief about their probability of occurrence before observing the data point on which inference needs to be performed. In addition to the latent variables $\boldsymbol{x}$, the model includes observable random variables $\boldsymbol{y}$. Given the priors $p(\boldsymbol{x})$ and the generative model that produces $\boldsymbol{y}$ from $\boldsymbol{x}$, the simulator implicitly specifies a joint distribution $p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{x})\, p(\boldsymbol{y}|\boldsymbol{x})$. Using an observation model defined through the specification of the likelihood $p(\boldsymbol{y}|\boldsymbol{x})$, families of inference algorithms (e.g., Markov chain Monte Carlo, importance sampling, variational inference) allow to automatically produce posterior distributions $p(\boldsymbol{x}|\boldsymbol{y})$, which describe values of $\boldsymbol{x}$ that could have generated the observed data $\boldsymbol{y}$, therefore explaining the observed data under the model specified by the probabilistic program. Crucially, the joint generative model $p(\boldsymbol{x}, \boldsymbol{y})$ can also be used to generate synthetic data instances $\boldsymbol{x}_i, \boldsymbol{y}_i \sim p(\boldsymbol{x}|\boldsymbol{y})$ sampled from the generative model.

3. METHODOLOGY

3.1. Bayesian ML for spacecraft conjunctions

In previous work [37] we discuss that approaching the spacecraft collision avoidance problem as discriminative

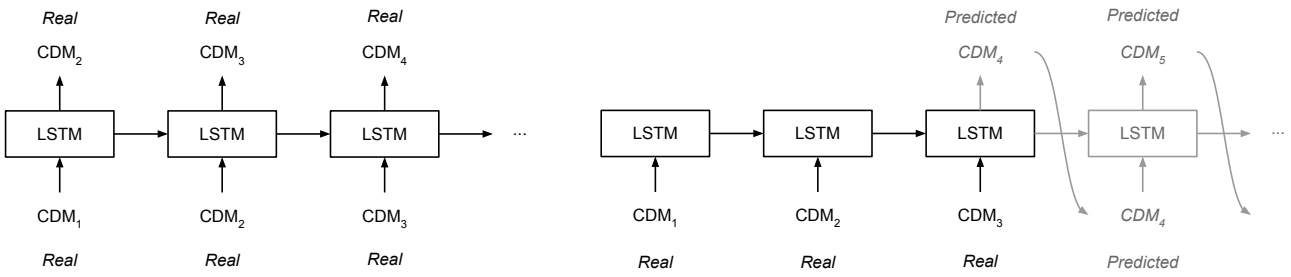

Figure 2. Left: *At training time the LSTM is optimized to reproduce the time-shifted CDM sequences in training data.* Right: *At test time, given a sequence of CDMs for the current event, the trained LSTM is used to predict future CDMs that characterise the event evolution.*

binary-classification formulation in ML (i.e., high- vs low-risk events) has various shortcomings (e.g., training data class imbalance, scarce interpretability of the output). For this reason, we focus our attention on the generative modeling of the problem, in other words we would like to deliver ML methods that describe the temporal evolution of a sequence of CDMs given all previously observed CDMs. More specifically, we train a special kind of recurrent neural network, a long short-term memory (LSTM) network [18] to predict the features and time of arrival of the next CDM given all previous CDMs in an evolving conjunction event (Figure 2). This setup is analogous to statistical language models in ML [32, 23] that predict the probability of the next symbol (e.g., a character or a whole word) in a given language, conditioned on all the symbols seen up to a given point. In order to quantify the uncertainty about the predicted values, we set up this network as a Bayesian neural network and apply Monte Carlo dropout at test time.

3.2. A generative probabilistic model for spacecraft orbits, collisions, and CDM generation

As mentioned previously and discussed in previous work [1], a pivotal component of the probabilistic programming approach is the definition and construction of a stochastic generative model. In this case, the purpose of the model is not only to mimic conjunction events, but also to simulate the CDM generation process. Once such a model is calibrated, it can then be used for either data generation or inference purposes. Here we briefly detail the construction of such a model and its key elements. The first essential component of the model are the priors: these are probability density functions describing the current resident space objects characteristics (e.g., mass, area) and orbits (e.g., orbital elements). We constructed these distributions based on real data collected from the public USSTRATCOM catalog and ESA DISCOS database.[3] By sampling from these distributions, two objects (i.e., the target and the chaser) are instantiated, and their initial *two-line elements* (TLEs) are extracted [35]. Once this is done, the orbits of the objects are propagated forward in time using an orbital propagator to produce ground-truth orbit trajectories. We used SGP4 propagator for preliminary experiments and design of the software, a fast low-precision propagator often used for preliminary analysis of orbital trajectories in low Earth orbit [19]. Having propagated the orbits, a check is made if a conjunction warning is to be triggered. If that is the case, we flag the event as a conjunction event and we simulate the ground observation part, by generating radar measurements, propagating them until TCA via Monte Carlo propagation, and generating CDMs associated with each observation. Both the timing of these observations and the measurement errors have been calibrated on the Kelvins dataset. In particular, the observation times have been formulated as probability distributions and have been calibrated to match Kelvins dataset distributions, whereas measurement errors were calibrated by tuning their values until the propagated covariance distributions at TCA were matching the ones in the Kelvins dataset.

[3] `https://discosweb.esoc.esa.int/`

4. SOFTWARE ARCHITECTURE

Kessler is written in Python 3 and made available through a GitHub repository.[4] Kessler software architecture consists of several components. The first one covers the importing, exporting, plotting and analysis functionalities, where CDMs can be loaded and analyzed both graphically and statistically. Then, two other main functionalities are present: a ML module where Bayesian LSTMs can be trained using a dataset of existing CDMs and subsequently used to make predictions of future CDMs in new conjunction events at test time; and a probabilistic programming module that provides a generative model for conjunction events and CDMs. In this section, we will briefly detail these Kessler functionalities and discuss their usage.

4.1. CDM importing and exporting, analysis, and visualization

Kessler offers two main possibilities to load CDMs: either from their original *kvn* format, or from a pandas `DataFrame` object. Once loaded, CDMs are divided into conjunction events automatically and their units fulfill the CDM standard format [12]. Then, the software offers two main functionalities to plot the conjunction events (i.e., time series of CDMs):

[4] `https://github.com/kesslerlib/kessler`

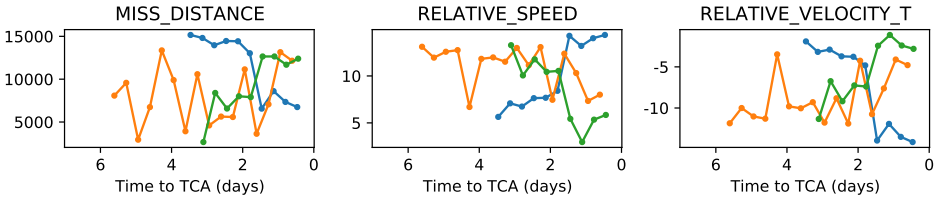

Figure 3. An example of `plot_features()` usage, for three events and with three different features plotted: relative speed, miss distance and along-track component of relative velocity.

Figure 4. An example of `plot_uncertainties()` usage, for three events.

- `plot_features()`: a function that takes as input the features to be visualized of the conjunction event(s) and plots them. An example for three events is shown in Figure 3.
- `plot_uncertainties()`: a function that plots all the covariance matrix elements for the considered events and for both the objects. An example of its usage is shown in Figure 4.

Moreover, the CDM content can also be manually inspected and processed by accessing the relevant CDMs from the event dataset object.

4.2. Machine learning module

As mentioned above, besides loading and plotting functionalities, the library also offers a ML module, where a Bayesian LSTM can be instantiated as an `LSTMPredictor` object and trained on datasets of CDMs. The ML implementations in Kessler are based on PyTorch[5] [36]. The `learn` function can be called by passing the relevant neural network training hyperparameters (e.g., batch size, learning rate). Once the model is learned from data, it can be used to predict future CDMs in new, previously unseen, conjunction events at test time. Kessler offers two different functions for doing this: `model.predict_event_step()` and `model.predict_event()`. While the former only predicts the next CDM, the latter predicts the entire CDM sequence, using the neural network's predicted CDM at each time step as an input for future time steps (Figure 2). Due to the Bayesian nature of the implemented stacked LSTM architecture, the network returns distributions rather than point predictions. In Figure 5, we show an example of both functions' usage.

[5] `https://pytorch.org/`

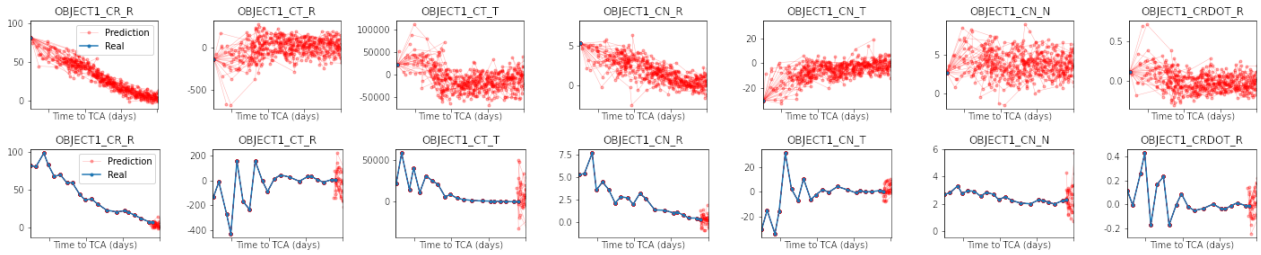

Figure 5. Top row: *usage of* `model.predict_event()`*, where the first CDM is used for predicting the whole event.* Bottom row: *usage of* `model.predict_event_step()`*, where the next CDM is predicted from a time series of CDMs.*

4.3. Probabilistic programming module

A key novelty of the Kessler package is its probabilistic programming module, which uses PyProb[6] [6], a universal probabilistic-programming framework based on PyTorch. This module can be used to generate synthetic conjunction events by simply importing the `Conjunction` object from Kessler, and by either sampling event realizations from the prior model and filtering the forward samples for only events that represent conjunctions or by generating each conjunction individually. This can be done by using two separate functions:

- `prior = model.prior(N)`: this samples the prior `N` times, therefore producing `N` ground truth orbits of target and chaser pairs. These events can then be filtered (using `model.filter()`) for only those that are conjunction events and thus contain associated synthetic CDMs. In Figure 6, we show several plots related to some prior runs where we show in orange the events that ended up in conjunctions and in blue those that did not.

- `model.get_conjunction()`: this automatically samples the prior distributions until a conjunction event is found, and returns the conjunction event, whose CDMs and features can then be analyzed, plotted and/or predicted.

As already discussed, this generative model can be used in PyProb probabilistic programming framework to perform posterior inference over model latents using `posterior = model.posterior(N, observation)`, conditioned on observed CDMs. For this, one has to choose the inference engine (e.g., importance sampling, Markov chain Monte Carlo) and the likelihood distributions that describe the probability of the observed data conditioned on the latents. In previous work [1], some experiments with these techniques have been presented and discussed.

[6] `https://github.com/pyprob/pyprob`

5. CONCLUSIONS

In this work we introduced Kessler, a new open-source project focused on delivering high-quality and tested ML techniques to the space community for applications in spacecraft collision avoidance. We would like to maintain and develop Kessler further as a state-of-the-art ML library with contributions from the larger ML and space communities. The community can leverage Kessler for several different purposes, and here we finish by listing some of its key use cases:

- Importing, analyzing, and visualizing CDM sequences;
- Training of ML models from private or public datasets of real CDMs in order to deploy these models to perform predictions at test time with new conjunction events with associated model uncertainties;
- Simulating conjunction events and generating realistic synthetic CDM datasets, which can be used as training data to enable further ML approaches;
- Performing Bayesian inference and therefore helping operators/users to determine key variables that lead to conjunction events and make reliable predictions (with associated uncertainties);
- Providing an open-source framework to host future risk assessment approaches based on ML and Bayesian inference.

ACKNOWLEDGMENTS

This work has been enabled by Frontier Development Lab (FDL) Europe, a public–private partnership between the European Space Agency (ESA), Trillium Technologies and the University of Oxford, and supported by Google Cloud. We would like to thank Dario Izzo and Moriba Jah for sharing their technical expertise and James Parr, Jodie Hughes, Leo Silverberg, and Alessandro Donati for their support. GA is supported by the Open Space Innovation Platform (OSIP) of the European Space Agency. AGB is supported by EPSRC/MURI grant EP/N019474/1 and by Lawrence Berkeley National Lab.

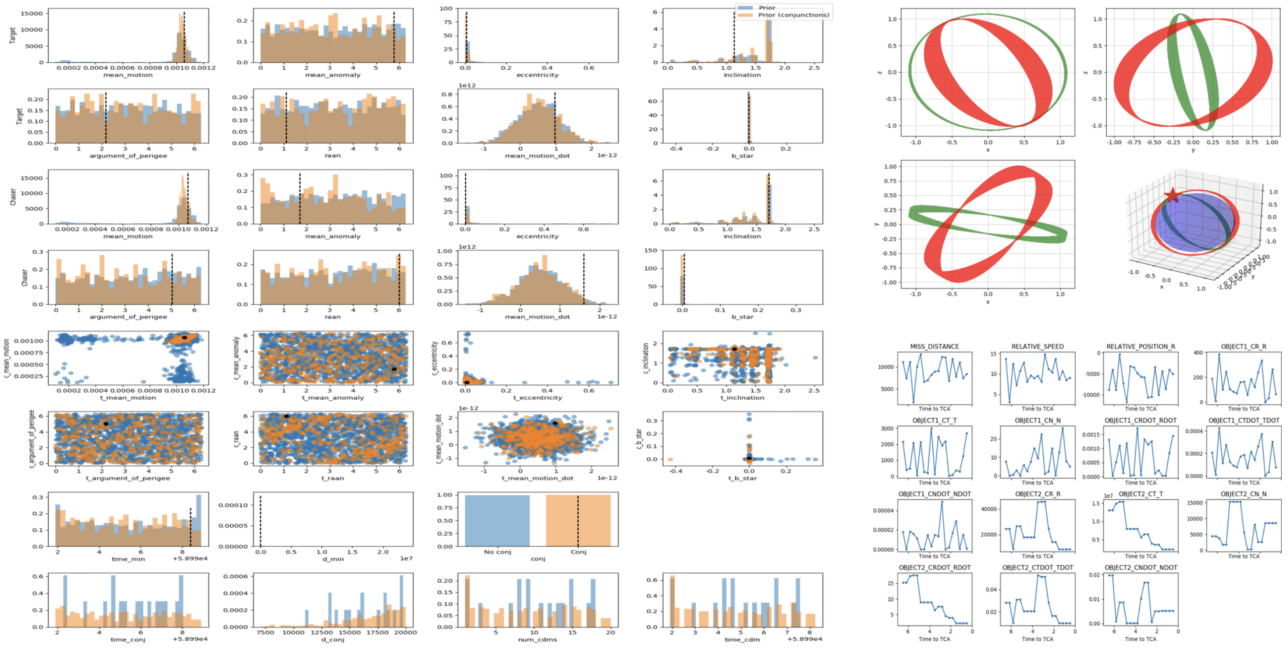

Figure 6. Several features of the sampled priors, including the distributions of the orbital elements of both target and chaser and the orbit geometry and some key CDMs features of one of the sampled conjunction events. Conjunction events are highlighted in orange and non-conjunction ones in blue.

REFERENCES

1. Giacomo Acciarini, Francesco Pinto, Sascha Metz, Sarah Boufelja, Sylvester Kaczmarek, Klaus Merz, José A Martinez-Heras, Francesca Letizia, Christopher Bridges, and Atılım Güneş Baydin. Spacecraft collision risk assessment with probabilistic programming. In *NeurIPS 2020: Machine Learning and the Physical Sciences Workshop*, 2020.
2. Nagavenkat Adurthi and Puneet Singla. Conjugate unscented transformation-based approach for accurate conjunction analysis. *Journal of Guidance, Control, and Dynamics*, 38(9):1642–1658, 2015.
3. Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *arXiv preprint arXiv:1606.06565*, 2016.
4. Luciano Anselmo and Carmen Pardini. Analysis of the consequences in low earth orbit of the collision between Cosmos 2251 and Iridium 33. In *Proceedings of the 21st International Symposium on Space Flight Dynamics*, pages 1–15. Centre nationale d'etudes spatiales Paris, France, 2009.
5. Atılım Güneş Baydin, Lukas Heinrich, Wahid Bhimji, Lei Shao, Saeid Naderiparizi, Andreas Munk, Jialin Liu, Bradley Gram-Hansen, Gilles Louppe, Lawrence Meadows, Philip Torr, Victor Lee, Prabhat, Kyle Cranmer, and Frank Wood. Efficient probabilistic inference in the quest for physics beyond the standard model. In *Advances in Neural Information Processing Systems 33 (NeurIPS)*, 2019.
6. Atılım Güneş Baydin, Lei Shao, Wahid Bhimji, Lukas Heinrich, Lawrence F. Meadows, Jialin Liu, Andreas Munk, Saeid Naderiparizi, Bradley Gram-Hansen, Gilles Louppe, Mingfei Ma, Xiaohui Zhao, Philip Torr, Victor Lee, Kyle Cranmer, Prabhat, and Frank Wood. Etalumis: Bringing probabilistic programming to scientific simulators at scale. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '19, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450362290. doi: 10.1145/3295500.3356180. URL `https://doi.org/10.1145/3295500.3356180`.
7. Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pages 1613–1622. PMLR, 2015.
8. V Braun, T Flohrer, H Krag, K Merz, S Lemmens, B Bastida Virgili, and Q Funke. Operational support to collision avoidance activities by ESA's space debris office. *CEAS Space Journal*, 8(3):177–189, 2016.
9. Inter-Agency Space Debris Coordination Committee et al. IADC space debris mitigation guidelines. *IADC, September*, 2007.
10. Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. *Proceedings of the National Academy of Sciences*, 117(48): 30055–30062, 2020.
11. Tim Flohrer and Holger Krag. Space surveillance and tracking in ESA's SSA programme. In *Proceedings 7th European Conference on Space Debris, Darmstadt, Germany, https://conference. sdo. esoc. esa. int*, 2017.

12. The Consultative Committee for Space Data Systems (CCSDS). *Conjunction Data Message: Recommended Standard CCSDS 508.0-B-1*. 2013.
13. Eric B Ford. Quantifying the uncertainty in the orbits of extrasolar planets. *The Astronomical Journal*, 129 (3):1706, 2005.
14. Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059. PMLR, 2016.
15. Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep Learning*, volume 1. MIT Press Cambridge, 2016.
16. Andrew D Gordon, Thomas A Henzinger, Aditya V Nori, and Sriram K Rajamani. Probabilistic programming. In *Future of Software Engineering Proceedings*, pages 167–181. 2014.
17. Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. arxiv preprint arxiv: 12070580, 2012.
18. Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.
19. FR Hoots and RL Roehrich. Models for propagation of norad element sets, spacetrack report no. 3. *Department of Commerce, National Technical Information Service*, 9, 1980.
20. Brandon A Jones, Alireza Doostan, and George H Born. Nonlinear propagation of orbit uncertainty using non-intrusive polynomial chaos. *Journal of Guidance, Control, and Dynamics*, 36(2):430–444, 2013.
21. Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? *arXiv preprint arXiv:1703.04977*, 2017.
22. Donald J Kessler and Burton G Cour-Palais. Collision frequency of artificial satellites: The creation of a debris belt. *Journal of Geophysical Research: Space Physics*, 83(A6):2637–2646, 1978.
23. Yoon Kim, Yacine Jernite, David Sontag, and Alexander Rush. Character-aware neural language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.
24. Heiner Klinkrad. Space debris. *Encyclopedia of Aerospace Engineering*, 2010.
25. Heiner Klinkrad, P Beltrami, S Hauptmann, C Martin, H Sdunnus, H Stokes, R Walker, and J Wilkinson. The esa space debris mitigation handbook 2002. *Advances in Space Research*, 34(5):1251–1259, 2004.
26. Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *arXiv preprint arXiv:1612.01474*, 2016.
27. Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.
28. Stijn Lemmens and Francesca Letizia. ESA's annual space environment report. Technical Report GEN-DB-LOG-00288-OPS-SD, ESA Space Debris Office, 2020.
29. J Liou and Nicholas L Johnson. Risks in space from orbiting debris. *Science*, 311(5759):340, 2006.
30. J-C Liou and Nicholas L Johnson. Instability of the present leo satellite populations. *Advances in Space Research*, 41(7):1046–1053, 2008.
31. Molly K Macauley. The economics of space debris: Estimating the costs and benefits of debris mitigation. *Acta Astronautica*, 115:160–164, 2015.
32. Gábor Melis, Chris Dyer, and Phil Blunsom. On the state of the art of evaluation in neural language models. *arXiv preprint arXiv:1707.05589*, 2017.
33. K Merz, B Bastida Virgili, V Braun, T Flohrer, Q Funke, H Krag, S Lemmens, and J Siminski. Current collision avoidance service by ESA's space debris office. In *Proceedings 7th European Conference on Space Debris, Darmstadt, Germany*, pages 18–21, 2017.
34. Theodore J Muelhaupt, Marlon E Sorge, Jamie Morin, and Robert S Wilson. Space traffic management in the new space era. *Journal of Space Safety Engineering*, 6(2):80–87, 2019.
35. NASA. Definition of two-line element set coordinate system, 2011. `https://spaceflight.nasa.gov/realdata/sightings/SSapplications/Post/JavaSSOP/SSOP_Help/tle_def.html`.
36. Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. 2017.
37. Francesco Pinto, Giacomo Acciarini, Sascha Metz, Sarah Boufelja, Sylvester Kaczmarek, Klaus Merz, José A Martinez-Heras, Francesca Letizia, Christopher Bridges, and Atılım Güneş Baydin. Towards automated satellite conjunction management with bayesian deep learning. In *NeurIPS 2020: AI for Earth Sciences Workshop*, 2020.
38. Jonas Radtke, Christopher Kebschull, and Enrico Stoll. Interactions of the space debris environment with mega constellations—using the example of the oneweb constellation. *Acta Astronautica*, 131:55–68, 2017.
39. Zhen-Jiang Sun, Ya-Zhong Luo, Pierluigi Di Lizia, and Franco Bernelli Zazzera. Nonlinear orbital uncertainty propagation with differential algebra and Gaussian mixture model. *SCIENCE CHINA Physics, Mechanics & Astronomy*, 62(3):34511, 2019.
40. Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

41. Thomas Uriot, Dario Izzo, Jose Martinez-Heras, Francesca Letizia, Jan Siminski, and Klaus Merz. Collision avoidance challenge dataset, January 2021. URL https://doi.org/10.5281/zenodo.4463683.

42. Thomas Uriot, Dario Izzo, Luis Simoes, Rasit Abay, Nils Einecke, Sven Rebhan, Jose Martinez-Heras, Francesca Letizia, Jan Siminski, and Klaus Merz. Spacecraft collision avoidance challenge: Design and results of a machine learning competition. *Astrodynamics*, 2021.

43. David A Vallado. *Fundamentals of astrodynamics and applications*, volume 12. Springer Science & Business Media, 2001.

44. Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, and Frank Wood. An introduction to probabilistic programming. *arXiv preprint arXiv:1809.10756*, 2018.

45. Massimiliano Vasile, Carlos Ortega Absil, and Annalisa Riccardi. Set propagation in dynamical systems with generalised polynomial algebra and its computational complexity. *Communications in Nonlinear Science and Numerical Simulation*, 75:22–49, 2019.

46. R Walker, CE Martin, PH Stokes, JE Wilkinson, and H Klinkrad. Analysis of the effectiveness of space debris mitigation measures using the delta model. *Advances in Space Research*, 28(9):1437–1445, 2001.

47. Andrew Gordon Wilson. The case for Bayesian deep learning. *arXiv preprint arXiv:2001.10995*, 2020.