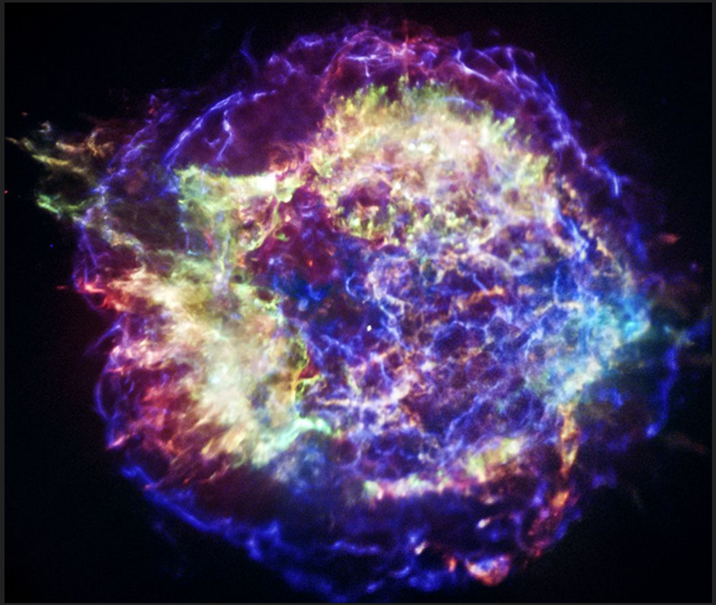
Simulation-based inference (sbi) for pulsar population synthesis

Dr. Vanessa Graber (graber@ice.csic.es)

in collaboration with Michele Ronchi, Celsa Pardo Araujo, and Nanda Rea

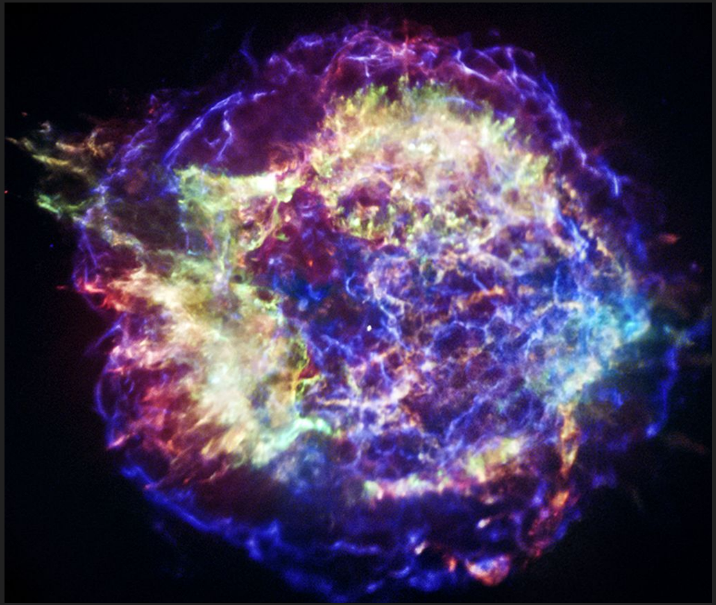
University of East Anglia, May 9th 2023



Cassiopeia A supernova remnant (credit: NASA/CXC/SAO)

Outline

- Neutron stars
- Pulsar population synthesis
- Machine learning and sbi
- Inference results
- Outlook



Cassiopeia A supernova remnant
(credit: NASA/CXC/SAO)

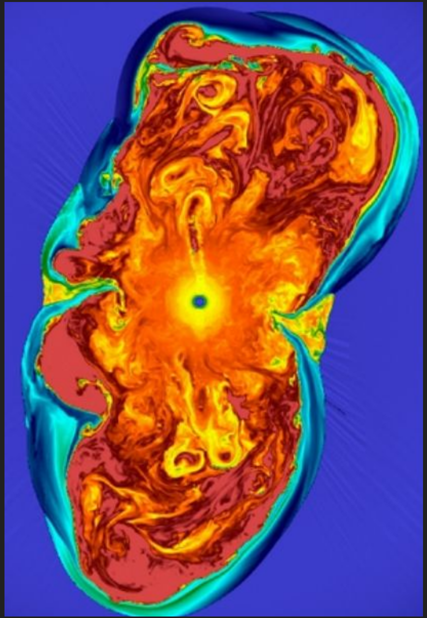
graber@ice.csic.es

Neutron-star formation

- Neutron stars are one of three types of **compact remnants**, created during the **final stages of stellar evolution**.

- When a **massive star of 8 - 25 solar masses** runs out of fuel, it collapses under its own gravitational attraction and **explodes in a supernova**.

- During the collapse, **electron capture** processes ($p + e^- \rightarrow n + \nu_e$) produce (a lot of) neutrons.



Snapshot of a 3D core-collapse supernova simulation (Mösta et al., 2014)

mass: 1.2 - 2.1 $M_{\odot}$

radius: 9 - 15 km

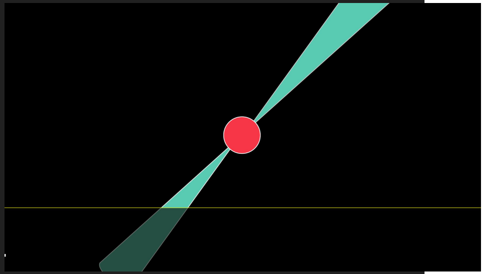
density: 10^{15} g/cm^3

Lighthouse radiation

- Neutron stars have **extreme magnetic fields** between 10^8 - 10^{15} G. For comparison, the Earth's magnetic field is 0.5 G.

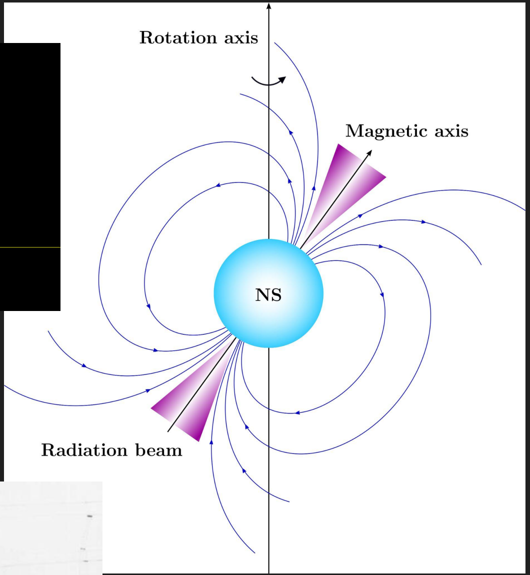
- Because rotation and magnetic axes are misaligned, neutron stars emit radio beams **like a lighthouse**.

- These pulses can be observed with radio telescopes. This is how neutron stars were first detected and why we call them **pulsars**.



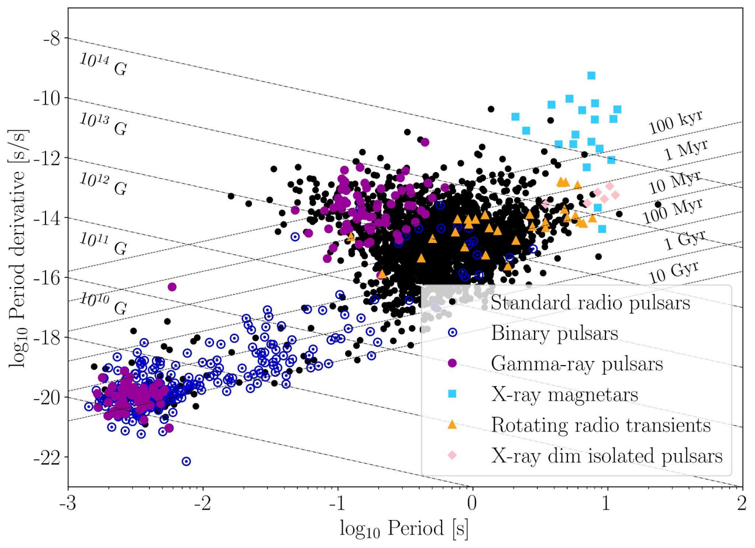
Credit: J. Christiansen

Sketch of the neutron-star exterior.



Dame Jocelyn Bell Burnell in front of her radio telescope in Cambridge, UK.

The neutron-star zoo



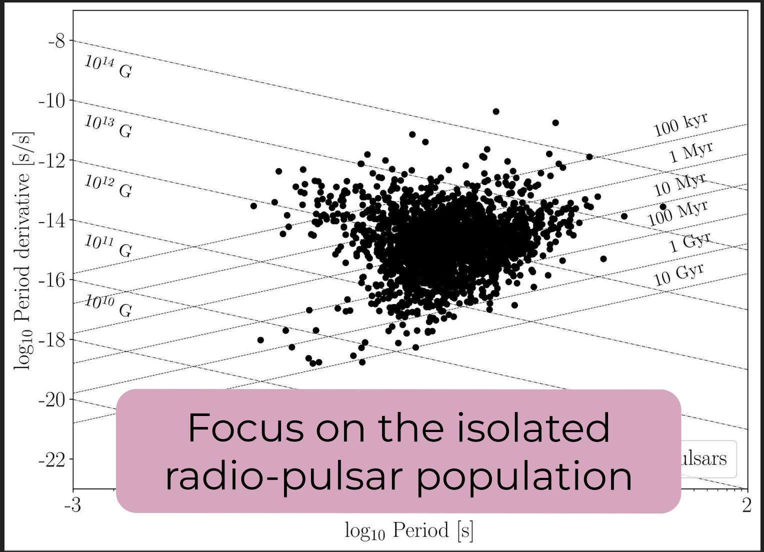
Period period-derivative plane for the pulsar population. Data taken from the ATNF Pulsar Catalogue (Manchester et al., 2005)

- Pulsars are **very precise clocks** and we time their pulses to **measure rotation periods P and derivatives $\dot{P}$**.
- We now observe neutron stars as pulsars **across** the **electromagnetic spectrum**.

~ **3,000 pulsars** are known to date

- Grouping neutron stars in the **$P\dot{P}$-plane** according to their observed properties serves as a diagnostic tool to **identify different neutron-star classes**.

The neutron-star zoo



Period period-derivative plane for the pulsar population. Data taken from the ATNF Pulsar Catalogue (Manchester et al., 2005)

- Pulsars are **very precise clocks** and we time their pulses to **measure rotation periods P and derivatives $\dot{P}$**.

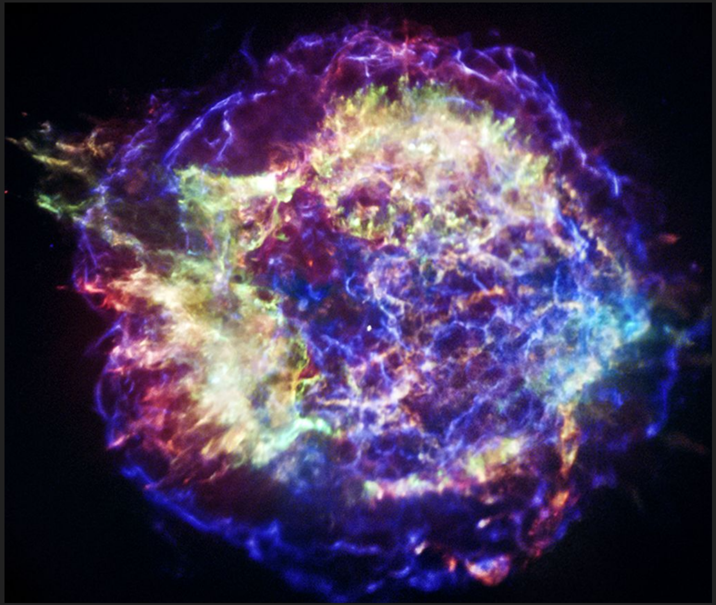
- We now observe neutron stars as pulsars **across** the **electromagnetic spectrum**.

~ **3,000 pulsars** are known to date

- Grouping neutron stars in the **$P\dot{P}$-plane** according to their observed properties serves as a diagnostic tool to **identify different neutron-star classes**.

Outline

- Neutron stars
- **Pulsar population synthesis**
- Machine learning and sbi
- Inference results
- Outlook



Cassiopeia A supernova remnant
(credit: NASA/CXC/SAO)

graber@ice.csic.es

General idea

- We can estimate the **total number of neutron stars in our Galaxy**

CC supernova rate: ~ 2 per century × **Galaxy age:** ~ 13.6 billion years = **NS number:** ~ 2.8×10^8

- We only **detect** a very **small fraction** of all neutron stars. Population synthesis bridges this gap focusing on the full population of neutron stars (e.g. Faucher-Giguère & Kaspi 2006, Lorimer et al. 2006, Gullón et al. 2014, Cieślar et al. 2020):

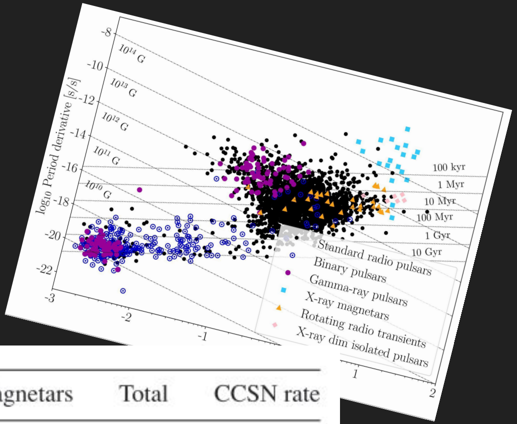
model **birth properties** with Monte-Carlo approach → **evolve properties** forward in time → **apply filters** to mimic observational biases/limits → **compare** mock simulations **to observations** to constrain input

Goals

- Population synthesis allows us to **constrain the natal properties** of neutron stars and their **birth rates**.

- This is for example **relevant for**:
    - Massive star evolution
    - Gamma-ray bursts
    - Fast-radio bursts
    - Peculiar supernovae

Estimated Galactic core-collapse supernova rate and birth rates for different pulsar classes (Keane & Kramer, 2008).



PSRs	RRATs	XDINSs	Magnetars	Total	CCSN rate
2.8 ± 0.5	$5.6^{+4.3}_{-3.3}$	2.1 ± 1.0	$0.3^{+1.2}_{-0.2}$	$10.8^{+7.0}_{-5.0}$	1.9 ± 1.1
1.4 ± 0.2	$2.8^{+1.6}_{-1.6}$	2.1 ± 1.0	$0.3^{+1.2}_{-0.2}$	$6.6^{+4.0}_{-3.0}$	1.9 ± 1.1
1.1 ± 0.2	$2.2^{+1.7}_{-1.3}$	2.1 ± 1.0	$0.3^{+1.2}_{-0.2}$	$5.7^{+4.1}_{-2.7}$	1.9 ± 1.1
1.6 ± 0.3	$3.2^{+2.5}_{-1.9}$	2.1 ± 1.0	$0.3^{+1.2}_{-0.2}$	$7.2^{+5.0}_{-3.4}$	1.9 ± 1.1
1.1 ± 0.2	$2.2^{+1.7}_{-1.3}$	2.1 ± 1.0	$0.3^{+1.2}_{-0.2}$	$5.7^{+4.1}_{-2.7}$	1.9 ± 1.1

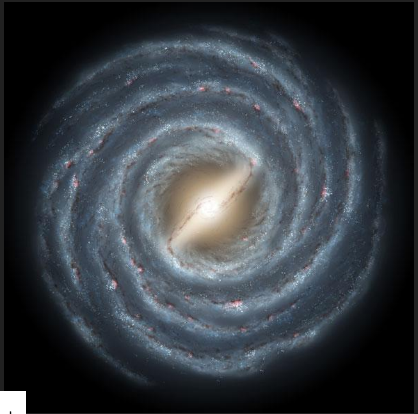
- We can also learn about **evolutionary links between different neutron-star classes** (e.g., Viganó et al., 2013). This is important because estimates for the **Galactic core-collapse supernova rate** are **insufficient** for to explain the independent formation of different classes of pulsars (Keane & Kramer, 2008).

Dynamical evolution I

- **Neutron stars are born in star-forming regions**, i.e., in the Galactic disk along the Milky Way's spiral arms, **and receive kicks** during the supernova explosions.

- We make the following assumptions:
  - Spiral-arm model (Yao et al., 2017) plus rigid rotation with T = 250 Myr
  - **Exponential disk model** with scale height h_c (Wainscoat et al., 1992)
  - Single-component **Maxwell kick-velocity distribution** with dispersion σ_k (Hobbs et al., 2005)
  - Galactic potential (Marchetti et al., 2019)

Artistic illustration of the Milky Way (credit: NASA JPL)



$$\mathcal{P}(z) = \frac{1}{h_c} e^{-\frac{|z|}{h_c}}$$

$$\mathcal{P}(v_{\mathrm{k}}) = \sqrt{\frac{2}{\pi}} \frac{v_{\mathrm{k}}^2}{\sigma_{\mathrm{k}}^3} e^{-\frac{v_{\mathrm{k}}^2}{2\sigma_{\mathrm{k}}^2}}$$

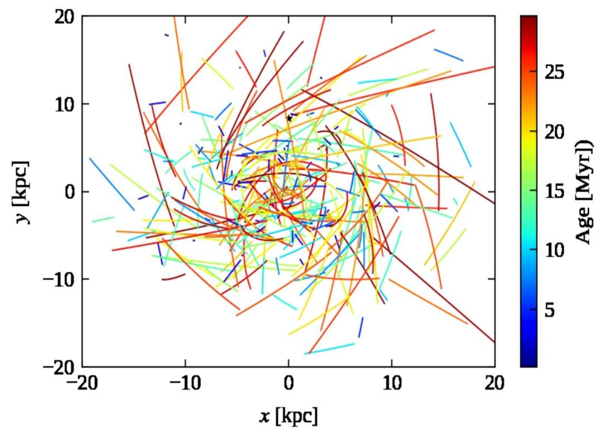
For Monte-Carlo approach, we **vary two uncertain parameters h_c and σ_k.**

Dynamical evolution II

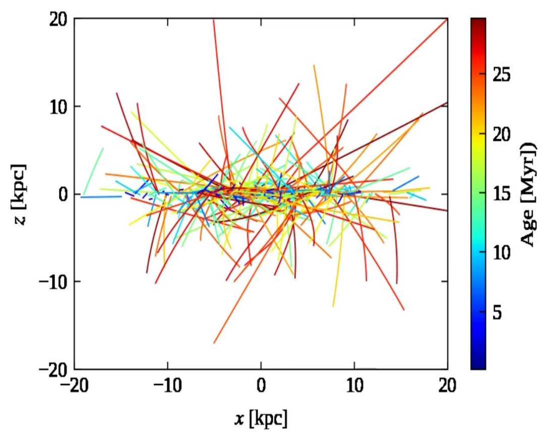
- For our Galactic model Φ_{MW}, we evolve the stars' position & velocity by **solving Newtonian equations of motion** in cylindrical galactocentric coordinates:

$$\ddot{\vec{r}} = -\vec{\nabla}\Phi_{MW}$$

Top view



Side view



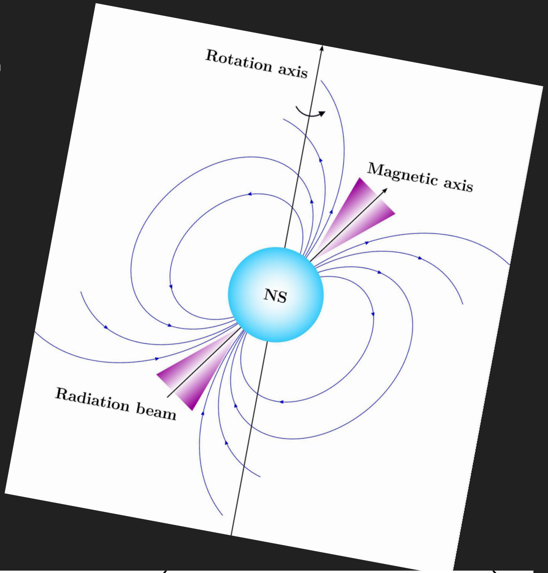
Galactic evolution tracks for h_c = 0.18 kpc, σ = 265 km/s.

Magneto-rotational evolution I

- The neutron-star magnetosphere exerts a **torque onto the star**. This causes **spin-down** and **alignment of the magnetic and rotation axes**.

- Neutron star **magnetic fields decay** due to the Hall effect and Ohmic dissipation in the outer stellar layer (crust) (e.g., Viganó et al., 2013 & 2021).

- We make the following assumptions:
  - **Initial periods** follow a log-normal with μ_P and σ_P (Igoshev et al., 2022)
  - **Initial fields** follow a log-normal with μ_B and σ_B (Gullón et al., 2014)



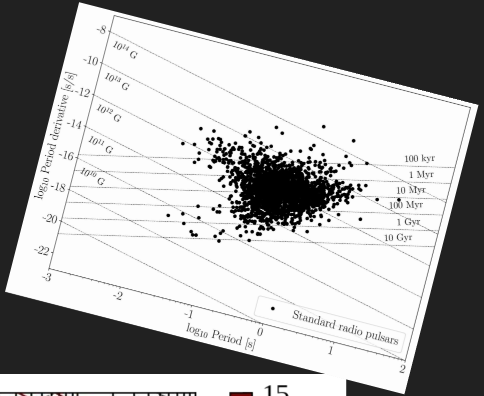
$$\mathcal{P}(P_0) = \frac{\log_{10}(\mathrm{e})}{\sqrt{2\pi} P_0 \sigma_P} \exp\left(- \frac{\left[\log_{10}(P_0) - \mu_P\right]^2}{2\sigma_P^2} \right)$$

Here, we **vary** the four uncertain parameters **μ_P, μ_B, σ_P and σ_B.**

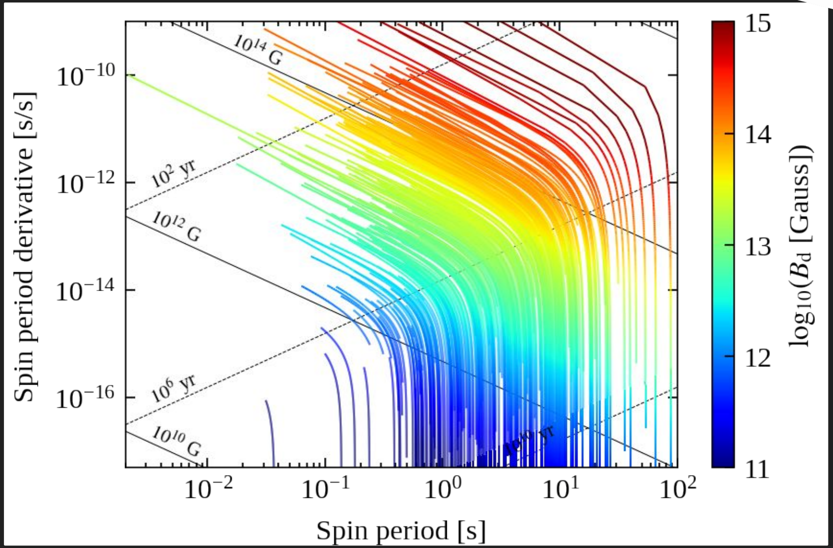
Magneto-rotational evolution II

- To model the neutron stars' magneto-rotational evolution, we numerically **solve three coupled ordinary differential equations** for the period, the misalignment angle and the magnetic field strength (Aguilera et al., 2008; Philippov et al. 2014).

- This allows us to follow the stars' P and P evolution in the $P\dot{P}$-plane.



$P\dot{P}$ evolution tracks for μ_P = -0.6 , σ_P = 0.3 , μ_B = 13.25 and σ_B = 0.75.



Radio emission and detection

- The stars' **rotational energy E_{rot}** is converted into coherent radio emission. We assume that the corresponding **radio luminosity L_{radio}** is proportional to the loss of E_{rot} (Faucher- Giguère & Kaspi, 2006; Gullón et al., 2014). L_0 is taken from observations.

$$L_{\rm radio} = L_0 \left(\frac{\dot{P}}{P^3} \right)^{1/2} \propto \dot{E}_{\rm dot}^{1/2}$$

- As **emission is beamed**, ~ 90% of pulsars do not point towards us. For those intercepting our line of sight, compute **radio flux S_{radio}** & **pulse width W**.

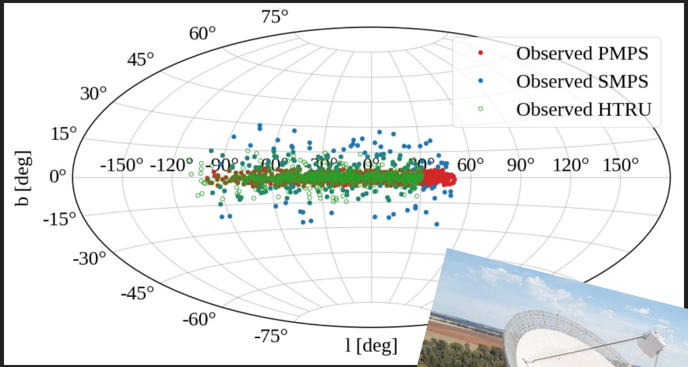
$$S_{\rm radio} = \frac{L_{\rm radio}}{\Omega_{\rm beam} d^2}$$

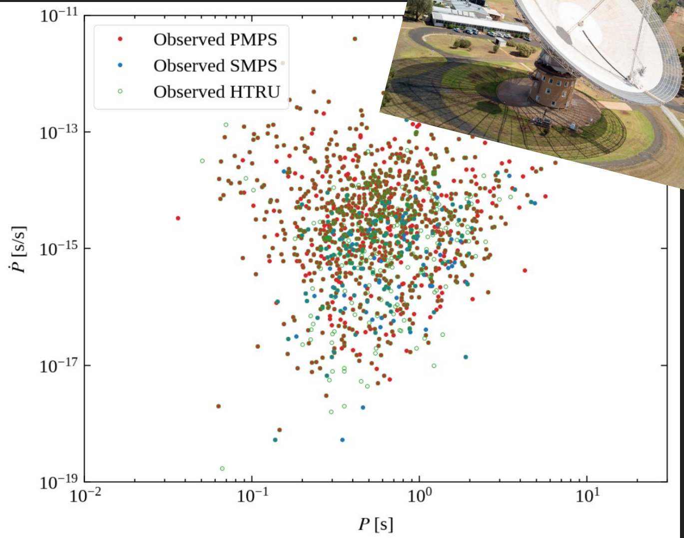
A pulsar counts as detected, if it **exceeds the sensitivity threshold** for a survey recorded with a specific radio telescope.

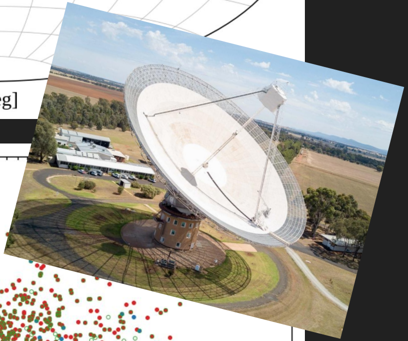
Three pulsar surveys

- We compare our simulated populations with three surveys from Murriyang (the Parkes Radio Telescope):
  - **Parkes Multibeam Pulsar Survey** (PMPS): *1,009 isolated pulsars*
  - **Swinburne Parkes Multibeam Pulsar Survey** (SMPS): *218 isol. p.*
  - **High Time Resolution Universe Survey** (HTRU): *1,023 isol. pulsars*

Can we constrain birth properties by looking at a current snapshot of the pulsar population?

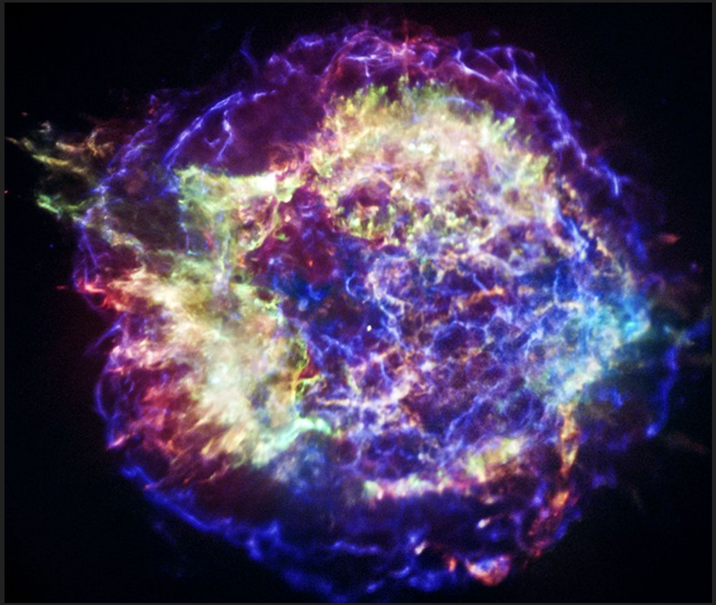






Outline

- Neutron stars
- Pulsar population synthesis
- Machine learning and sbi
- Inference results
- Outlook



Cassiopeia A supernova remnant
(credit: NASA/CXC/SAO)

graber@ice.csic.es

Comparing models and data

- Comparing observations to models and **constraining regions of the parameter space** that are **most probable given the data** is fundamental to many fields of science.

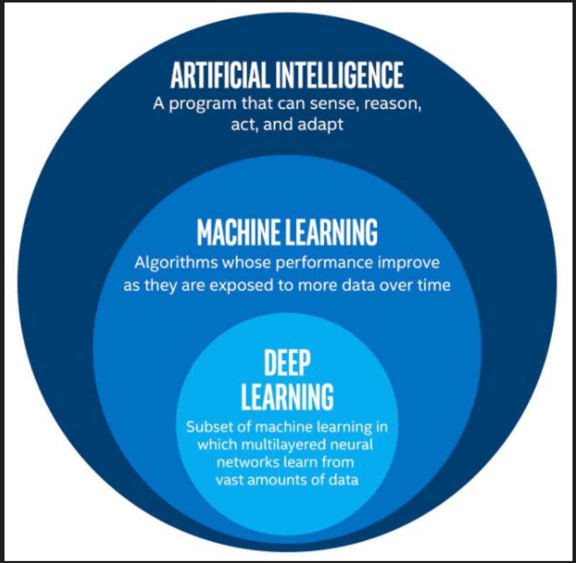
- Pulsar population synthesis is complex and has **many free parameters**. To compare synthetic simulations with observations, people have
    - Randomly sampled and then optimised 'by eye' (e.g., Gonthier et al., 2007)
    - Compared distributions of individual parameters using χ^2- and KS-tests (e.g., Narayan & Ostriker, 1990; Faucher-Giguère & Kaspi, 2006)
    - Used annealing methods for optimisation (Gullón et al., 2014)
    - Performed Bayesian inference for simplified models (Cieślar et al., 2020)

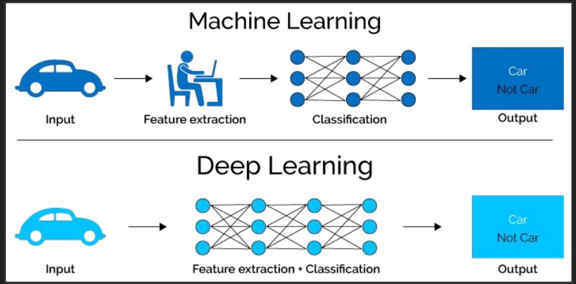
These methods do not scale well and are **difficult to use** with the **multi-dimensional data** produced in population synthesis.

Deep learning

- Deep Learning is a subfield of Artificial Intelli- gence and Machine Learning. It focuses on using **multi-layered neural networks** to learn from large datasets. Different to classi- cal ML approaches, deep learning does **not require external feature engineering**.

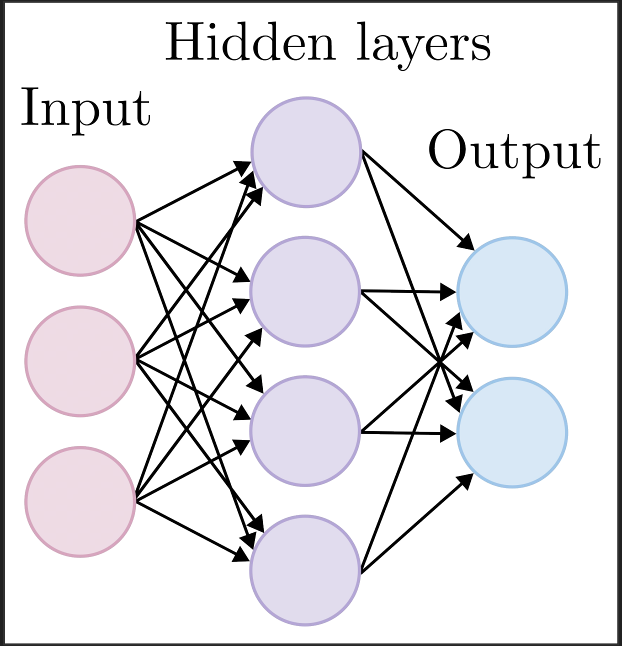
- **Recognising features in a hierarchical way** allows deep neural networks to model **complex non-linear relationships** for large input data. This makes deep learning powerful when **working with unstructured data such as images**, where the number of features / pixel can easily exceed millions.





Credit: www.bigdata-insider.de (top), Acheron Analytics (bottom)

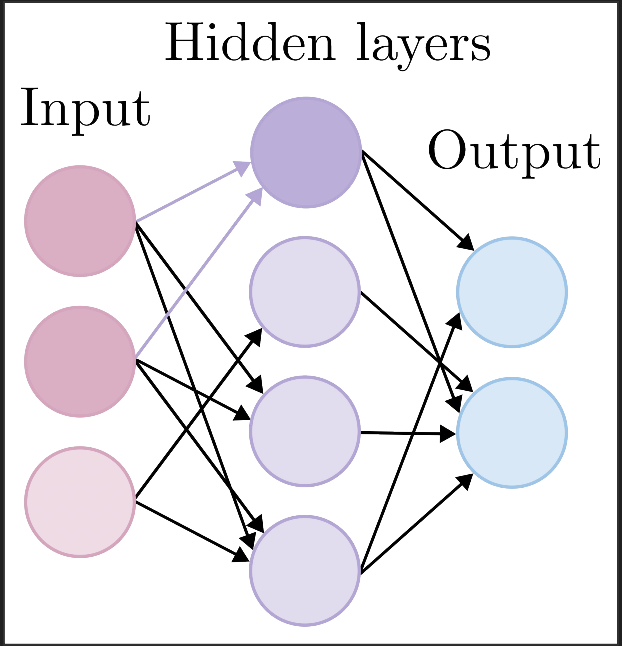
Convolutional neural networks (CNNs)



Sketch of a very simple
fully connected neural network.

- A neural network is composed of layers, which represent **stacks of neurons** (objects holding a single numerical value). Each layer encodes a simplified representation of the input data.
- A deep-learning **algorithm learns more and more about the input** as the data is passed through successive network layers.
- The **Multilayer Perceptron** is the simplest set-up where input and output are **fully connected**. In a CNN, not all nodes are connected, which **reduces the number of trainable parameters** and allows more flexibility for training.

Convolutional neural networks (CNNs)



Sketch of a very simple convolutional neural network.

- A neural network is composed of layers, which represent **stacks of neurons** (objects holding a single numerical value). Each layer encodes a simplified representation of the input data.
- A deep-learning **algorithm learns more and more about the input** as the data is passed through successive network layers.
- The **Multilayer Perceptron** is the simplest set-up where input and output are **fully connected**. In a CNN, not all nodes are connected, which **reduces the number of trainable parameters** and allows more flexibility for training.

Convolutional and max pooling layers

- Besides fully connected layers, CNNs are composed of two types of filters:

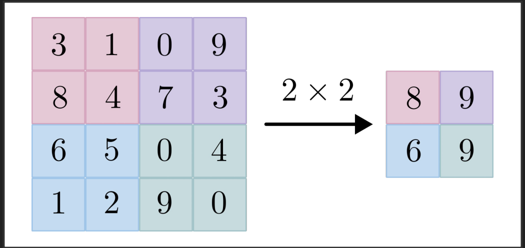
Convolutional filters

Source: Wikipedia

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

These filters recognise features, such as detecting edges of an object in an image.

Max-pooling layers



These filters extract the most relevant features, helping to speed up the training process.

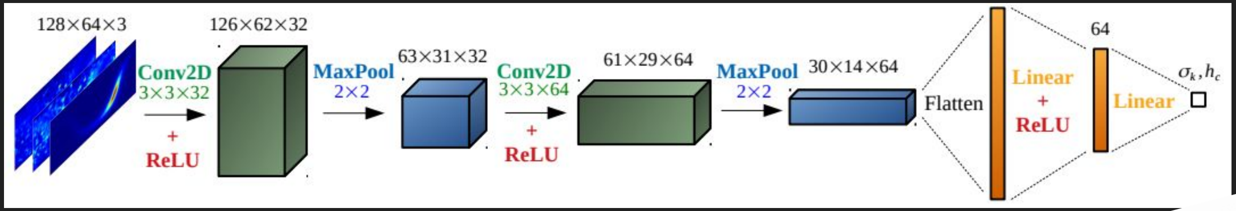
Proof of concept study I

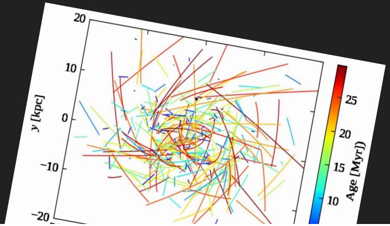
- In Ronchi et al. (2021), we focused on the dynamical evolution and **simulated a database of 128 x 128 (=16,384) synthetic neutron-star populations**.

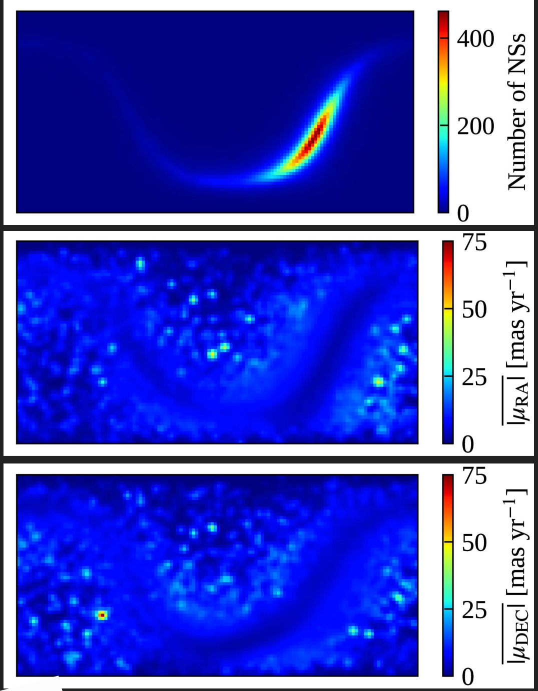
Vary **scale height h_c** in range [0.02-2] kpc

Vary **dispersion σ_k** of kick distribution between [1-700] km/s

- We **perform supervised ML** and train a CNN to extract labels h_c and σ from position / velocity maps:







Stellar density and velocity maps in ICRS coordinates.

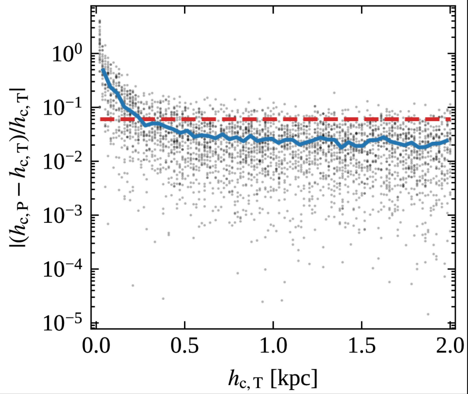
PyTorch

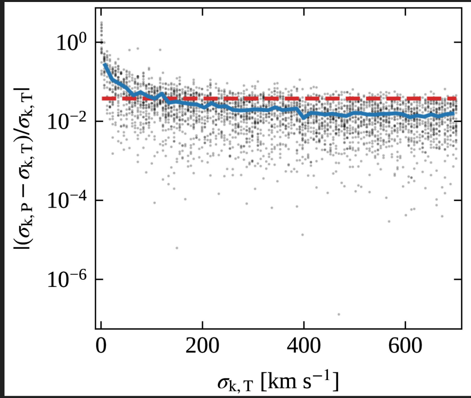
MAGNESIA erc CSIC IEEC

Proof of concept study II

- **Training info:** We use the root mean square error as the loss function and validation metric, Kaiming initialisation, Adam for gradient-descent optimisation, and apply a 80 / 20% split of the full dataset for training and validation.

- The **CNN recovers the input values** very well. To visualise this, we can look at the **relative error between target and predicted labels**

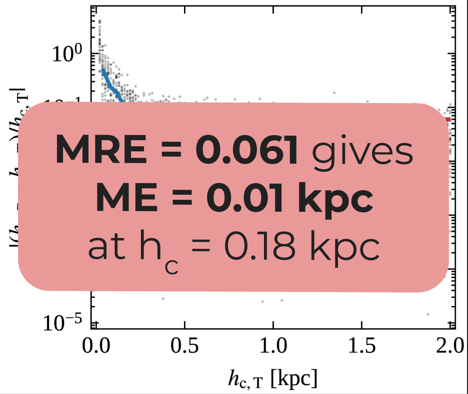




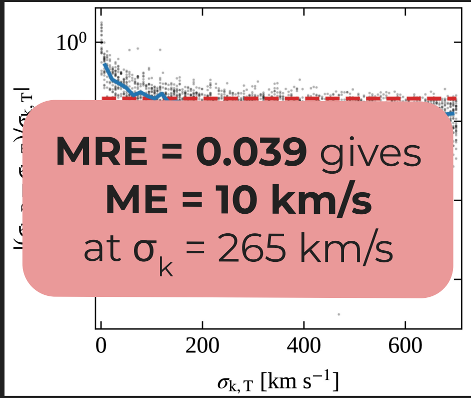
Proof of concept study II

- **Training info:** We use the root mean square error as the loss function and validation metric, Kaiming initialisation, Adam for gradient-descent optimi-sation, and apply a 80 / 20% split of the full dataset for training and validation.

- The **CNN recovers the input values** very well. To visualise this, we can look at the **relative error between target and predicted labels**



MRE = 0.061 gives
ME = 0.01 kpc
at h_c = 0.18 kpc



MRE = 0.039 gives
ME = 10 km/s
at σ_k = 265 km/s

We did **not include observational biases** and assumed all simulated stars are detectable! **216 pulsars have measured proper motions**, insuffi- cient for this precision.

Statistical inference

- Our initial study focused on **deducing point estimates**. However, we often do not require exact estimates but **knowledge of probable regions** is sufficient.

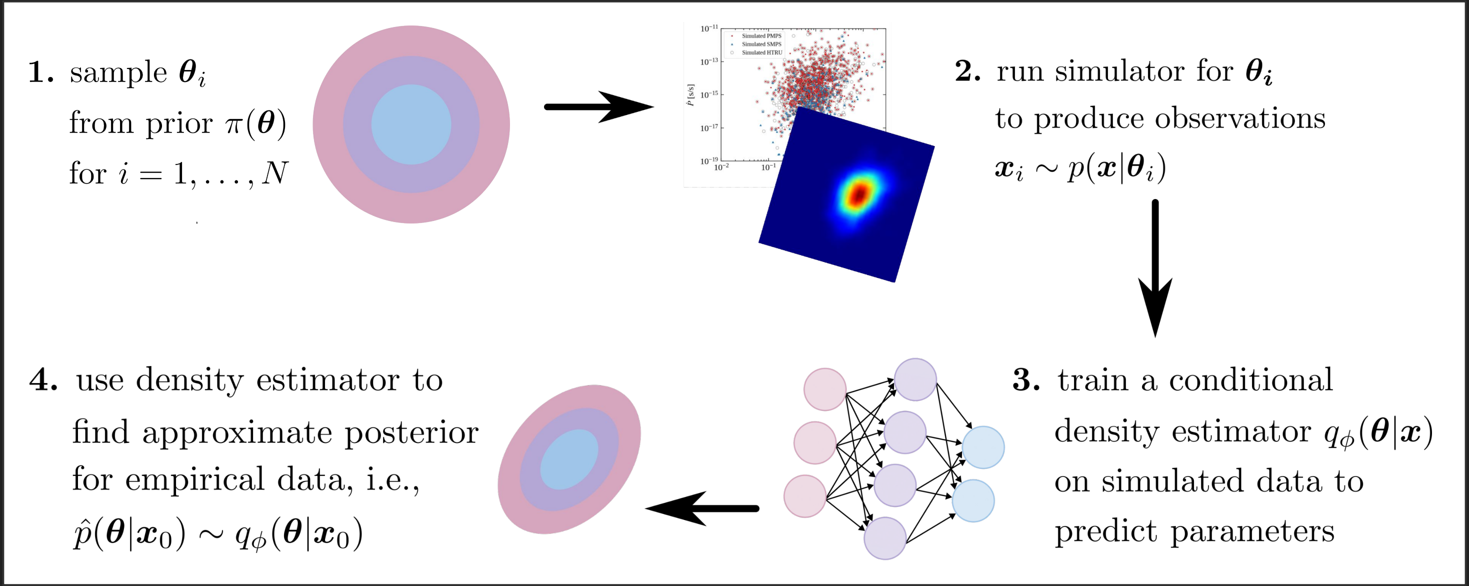
- This is where **Bayesian inference** comes in: based on some prior knowledge $\pi(\theta)$, a stochastic model and some observation x, we want to infer the most likely distribution $P(\theta|x)$ for our model parameters θ given the data x. This is **encoded in Bayes' Theorem**:

$$\underbrace{\mathcal{P}(\theta|x)}_{\text{posterior}} = \frac{\overbrace{\mathcal{P}(\theta)}^{\text{prior } \pi} \; \overbrace{\mathcal{P}(x|\theta)}^{\text{likelihood } \mathcal{L}}}{\underbrace{\mathcal{P}(x)}_{\text{evidence}}} = \frac{\mathcal{P}(\theta) \int \mathcal{P}(x, z|\theta)\mathrm{d}z}{\int \mathcal{P}(x|\theta')\mathcal{P}(\theta')\mathrm{d}\theta'}$$

For complex simulators, the **likelihood is defined implicitly and often intractable**. This is overcome with **simulation-based** (likelihood-free) **inference** (see e.g. Cranmer et al., 2020).

Simulation-based inference I

- To perform **Bayesian inference for any kind of (stochastic) forward model** (e.g. those specified by simulators), we use the following approach:



Simulation-based inference II

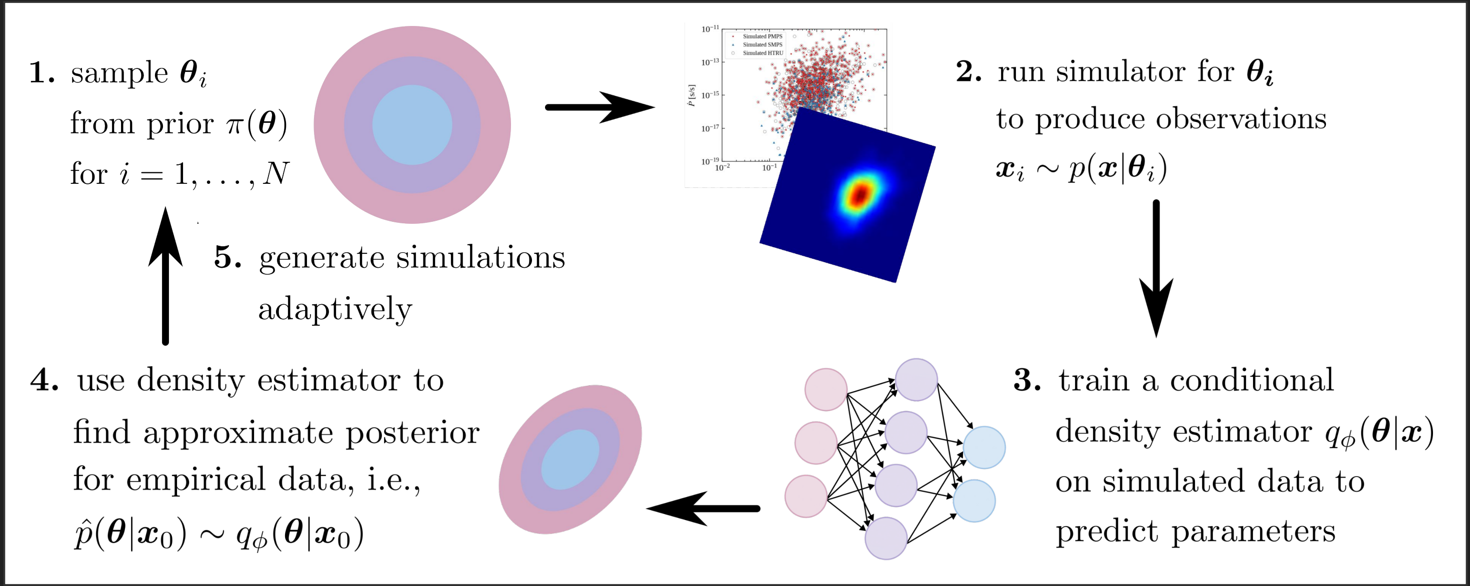
- Different approaches (all relying on deep learning) exist to **learn a probabilistic association** between the simulated data and the underlying parameters. These algorithms essentially focus on different pieces of Bayes' theorem:
  - Neural Posterior Estimation (NPE) (e.g., Papamakarios & Murray, 2016)
  - Neural Likelihood Estimation (NLE) (e.g., Papamakarios et al., 2019)
  - Neural Ratio Estimation (NRE) (e.g., Hermans et al., 2020; Delaunoy et al., 2022)

We focus on NPE. This allows us to **directly learn the posterior distribution**. In contrast, NLE and NRE need an extra (potentially time consuming) MCMC sampling step to construct a posterior.

- All methods exist in **sequential form** (SNPE, SNLE, SNRE), **which adds a fifth step to workflow**. Instead of sampling from the prior, we adaptively generate simulations from the posterior. This **typically requires fewer simulations**.

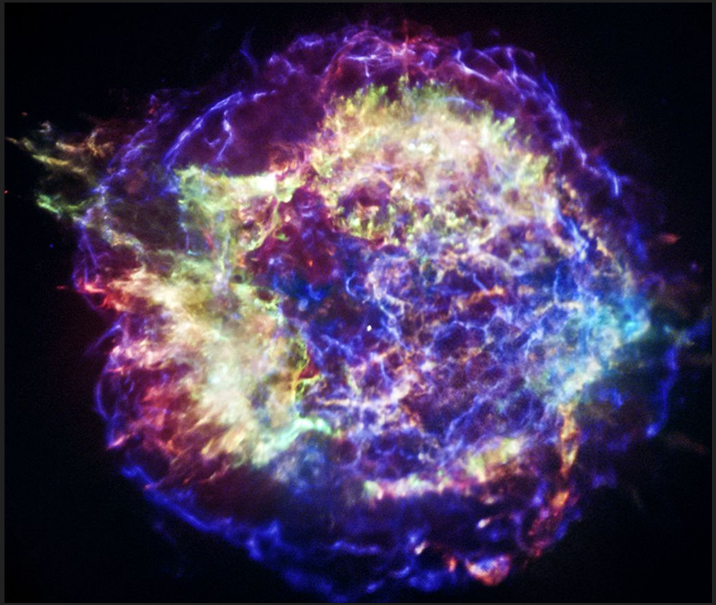
Simulation-based inference I

- To perform **Bayesian inference for any kind of (stochastic) forward model** (e.g. those specified by simulators), we use the following approach:



Outline

- Neutron stars
- Pulsar population synthesis
- Machine learning and sbi
- **Inference results**
- Outlook



Cassiopeia A supernova remnant
(credit: NASA/CXC/SAO)

Workflow

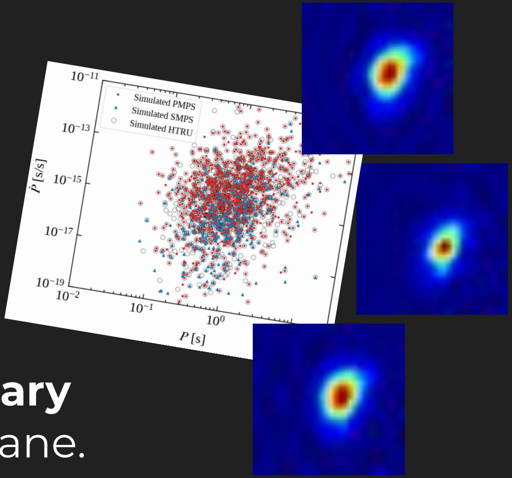
Varying the four parameters μ_P, μ_B, σ_P and σ_B, we simulate 360,000 synthetic populations over 6 weeks.

- With our complex population synthesis simulator, we fix the dynamics to a fiducial model and **focus on the magneto-rotational evolution**.
- From our simulated populations, we **generate summary statistics**: density maps for three surveys in the PP-plane.
- To perform the inference, we use the **PyTorch package sbi** (Tejero-Cantero et al., 2020; https://www.mackelab.org/sbi/). Our trainable neural network has two parts:
  - **CNN** (see Ronchi et al., 2021): compresses the data into a latent vector.
  - **Mixture density network**: our posterior is approximated by a mixture of 10 Gaussians components; we learn the means, stds and coefficients.
- We initialise the CNN with Kaiming, use 89% of data for training, 10% for validation and 1% for testing, set the batch size to 8, and learning rate to 5×10^{-4}.

MAGNESIA erc CSIC Consejo Superior de Investigaciones Científicas IEEC

graber@ice.csic.es

Workflow



- With our complex population synthesis simulator, we fix the dynamics to a fiducial model and **focus on the magneto-rotational evolution**.

- From our simulated populations, we **generate summary statistics**: density maps for three surveys in the PP-plane.

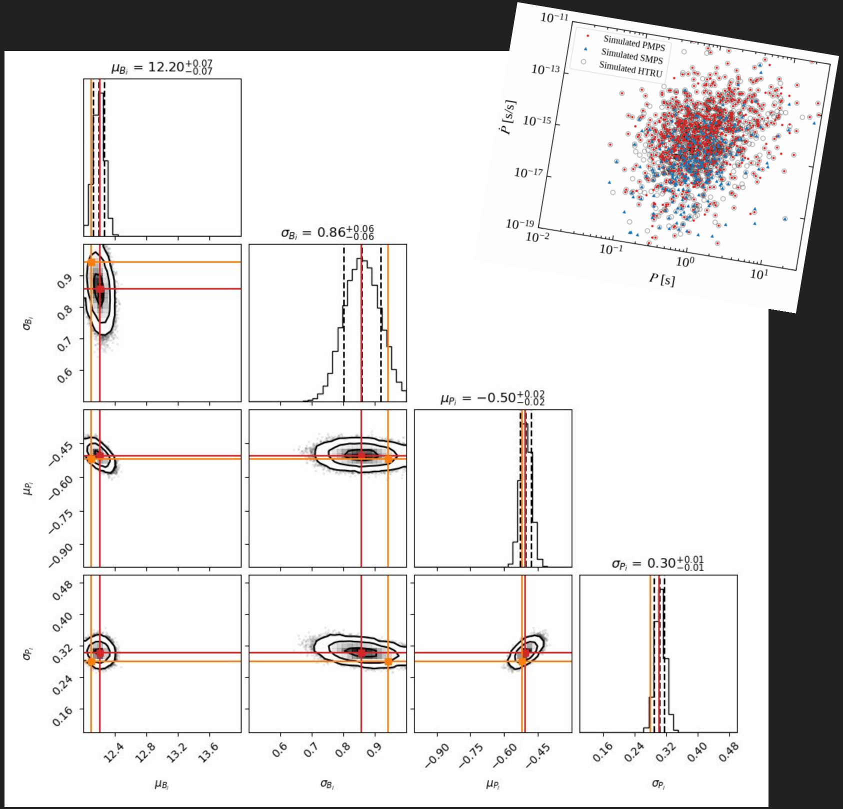
- To perform the inference, we use the **PyTorch package sbi** (Tejero-Cantero et al., 2020; https://www.mackelab.org/sbi/). Our trainable neural network has two parts:
  - **CNN** (see Ronchi et al., 2021): compresses the data into a latent vector.
  - **Mixture density network**: our posterior is approximated by a mixture of 10 Gaussians components; we learn the means, stds and coefficients.

- We initialise the CNN with Kaiming, use 89% of data for training, 10% for vali-dation and 1% for testing, set the batch size to 8, and learning rate to 5×10^{-4}.

Posterior distributions for test sample

- As our conditional density estimator is represented by a neural network, we can **directly evaluate the posterior distributions for a given observation**.

- We recover **narrow, well-defined posteriors** for all four parameters that contain the ground truth (parameters used for the forward simulation) at the 2σ levels.



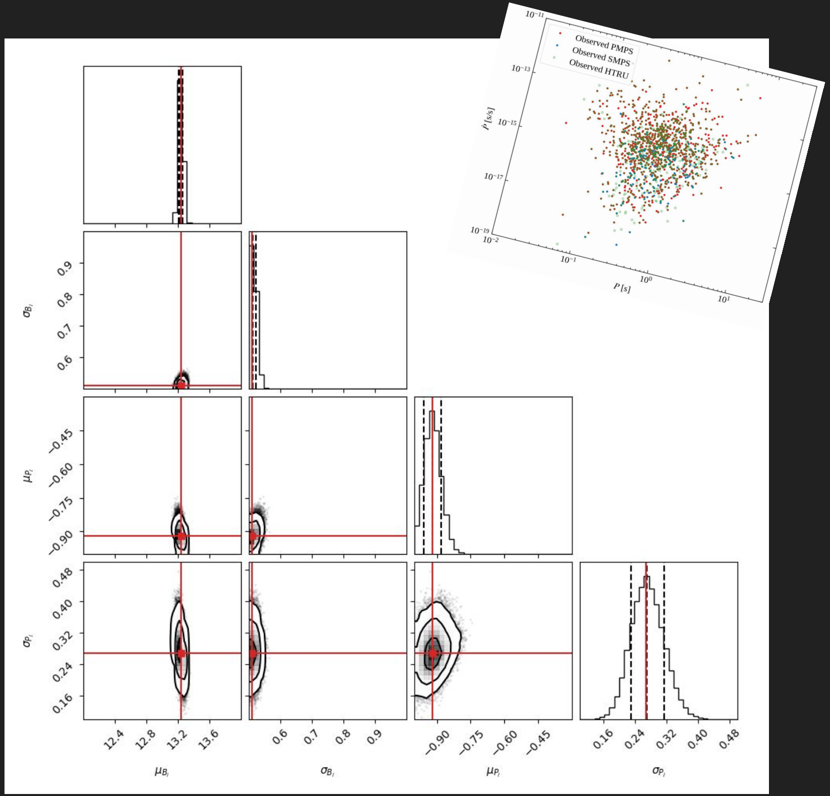
Posterior distributions for observed population

- With our optimised neural network, we can also **infer the posteriors** for the **pulsar population recorded in our three surveys** and recover the following constraints:

$$\mu_P = -0.92^{+0.04}_{-0.04} \qquad \sigma_P = 0.27^{+0.04}_{-0.04}$$

$$\mu_B = 13.23^{+0.03}_{-0.03} \qquad \sigma_B = 0.51^{+0.01}_{-0.01}$$

$$\mathcal{P}(P_0) = \frac{\log_{10}(\mathrm{e})}{\sqrt{2\pi}P_0\sigma_P}\exp\left(-\frac{[\log_{10}(P_0)-\mu_P]^2}{2\sigma_P^2}\right)$$



Posterior distributions for observed population

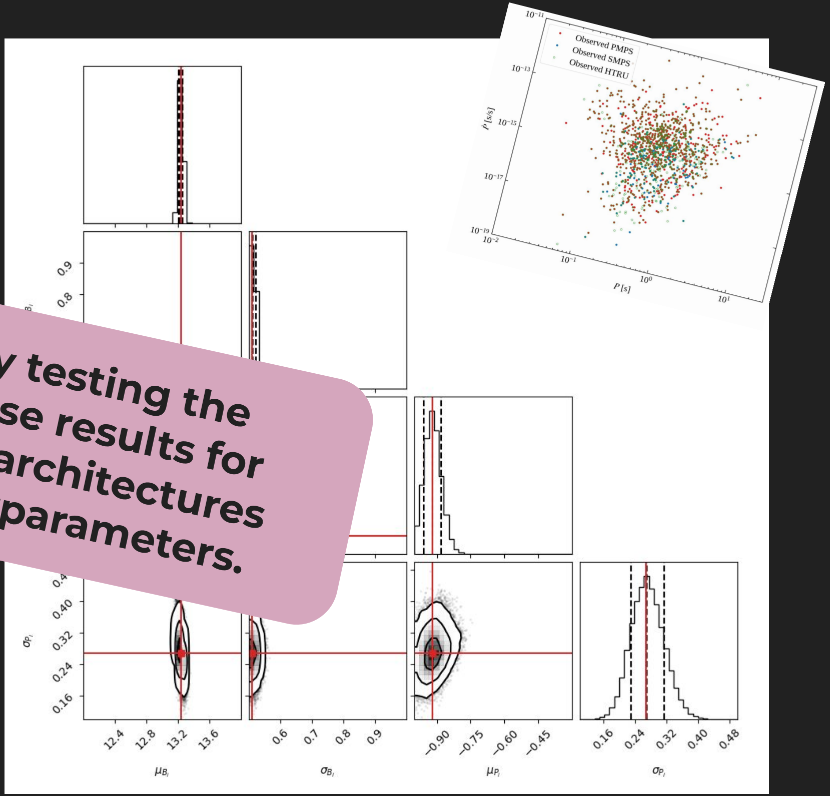
- With our optimised neural network, we can also **infer the posteriors** for the **pulsar population recorded in our three surveys** and following constrai

$$\mu_P = -0.92^{+0.08}_{-0.07}$$

$$\sigma_P = 0.27^{+0.09}_{-0.07}$$

$$\sigma_B = 0.51^{+0.02}_{-0.02}$$

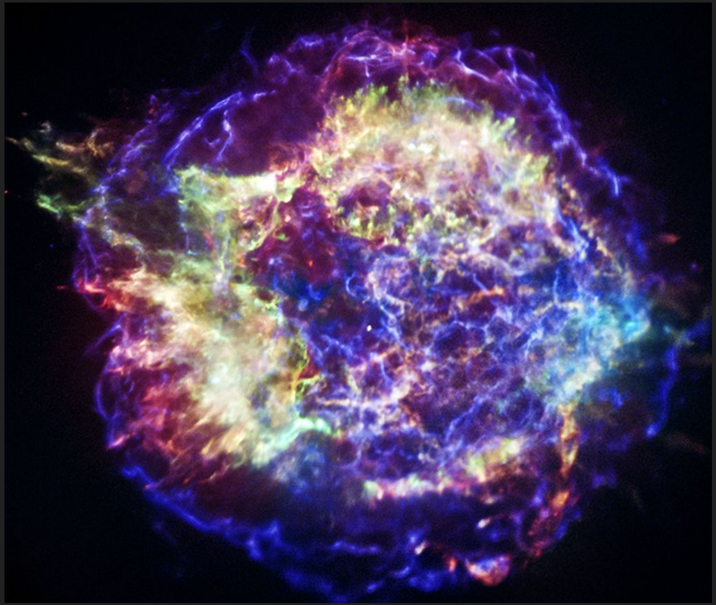
$$\mathcal{P}(P_0) = \frac{\log_{10}(\mathrm{e})}{\sqrt{2\pi}P_0\sigma_P}\exp\left(-\frac{\left[\log_{10}(P_0)-\mu_P\right]^2}{2\sigma_P^2}\right)$$



We are currently testing the robustness of these results for different network architectures and training hyperparameters.

Outline

- Neutron stars
- Pulsar population synthesis
- Machine learning and sbi
- Inference results
- **Summary and outlook**



Cassiopeia A supernova remnant
(credit: NASA/CXC/SAO)

graber@ice.csic.es

Take-home points

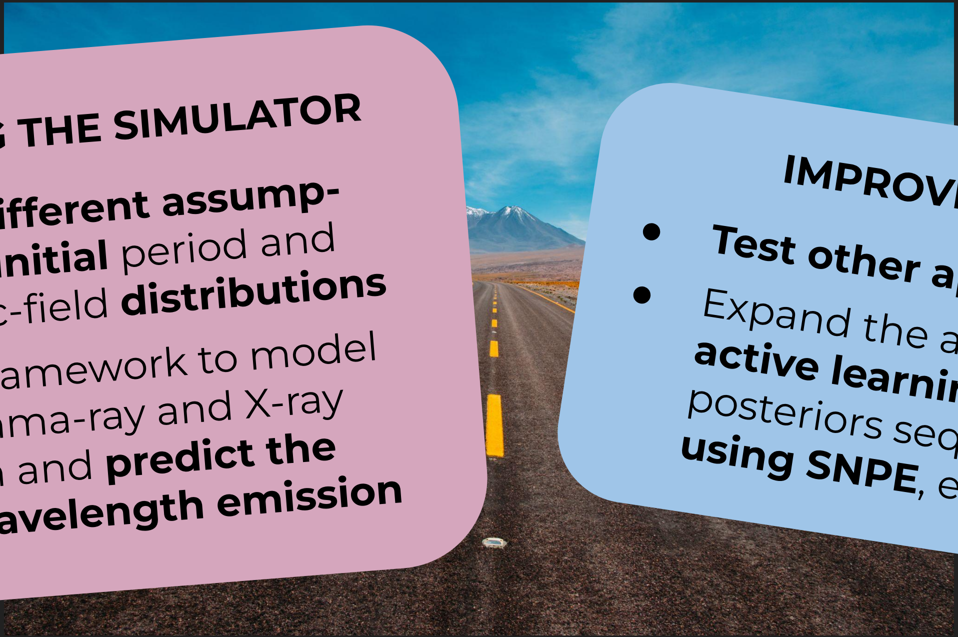
- Neutron stars are **compact remnants** that **emit pulsed radiation** across the electro-magnetic spectrum.
- **Standard radio pulsars** constitutes the largest class of observed neutron star.

- Pulsar **population synthesis** bridges gap between known pulsars and the invisible population.
- **It allows** us **to constrain birth rates** of different neutron star classes **and birth properties**.

- **Deep learning** with neural networks is ideal to **analyse high-dimensional astrophysical data**.
- **Simulation-based inference** has opened up the possibility for statistical inference **for complex simulators**.

Outlook

- There are **several directions** that we have started to look into:



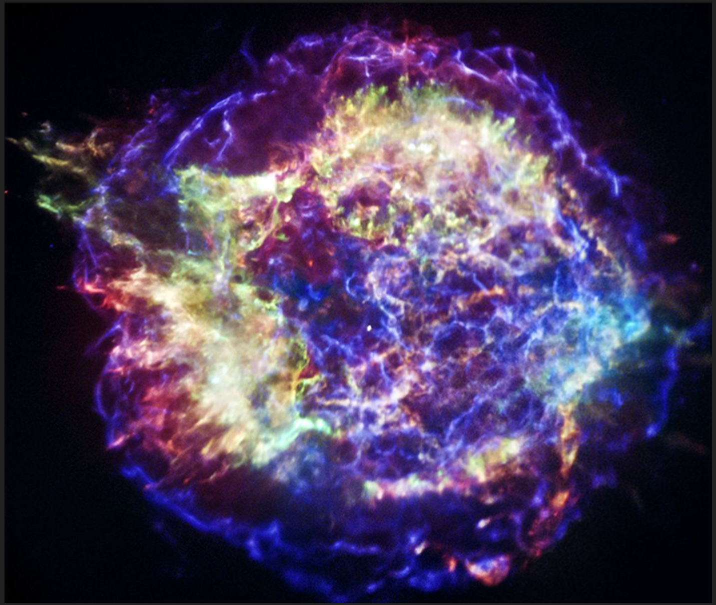
IMPROVING THE SIMULATOR

- Explore **different assumptions on initial** period and magnetic-field **distributions**
- Extend framework to model also gamma-ray and X-ray emission and **predict the multi-wavelength emission**

IMPROVING SBI

- **Test other approaches**
- Expand the approach to **active learning** and derive posteriors sequentially **using SNPE**, etc.

THANK YOU

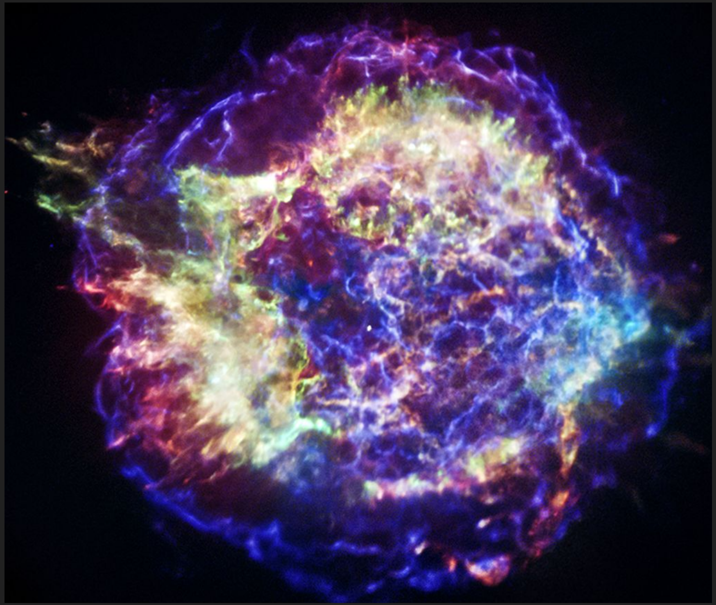


Cassiopeia A supernova remnant
(credit: NASA/CXC/SAO)

graber@ice.csic.es

Back-up

Slides



Cassiopeia A supernova remnant
(credit: NASA/CXC/SAO)

MAGNESIA erc CSIC Consejo Superior de Investigaciones Científicas IEEC

graber@ice.csic.es

Magneto-rotational evolution - analytics

- For a given initial rotation period, magnetic field strength and inclination angle (uniformly distributed along the sphere), the evolution of a neutron star is determined by **three ordinary differential equations**. The first two are

$$\dot{P} = \frac{\pi^2}{c^3} \frac{B^2 R^6}{IP} \left(\kappa_0 + \kappa_1 \sin^2 \chi \right) \qquad \dot{\chi} = -\frac{\pi^2}{c^3} \frac{B^2 R^6}{IP^2} \left(\kappa_2 \sin \chi \cos \chi \right)$$

  The κ are determined from simulations. For a **realistic pulsar magnetosphere** filled with plasma we have $\boldsymbol{\kappa_0 \simeq \kappa_1 \simeq \kappa_2 \simeq 1}$ (Spitkovsky 2006; Philippov et al., 2014).

- From the induction equation, we deduce for the B-field (Aguilera et al., 2008):

$$\dot{B} = -\frac{c^2 B}{4\pi\sigma L^2} - \frac{cB^2}{4\pi e n_e L^2} \quad \text{with} \quad \tau_{\mathrm{Ohm}} = \frac{4\pi\sigma L^2}{c^2} \sim 10^6 \text{ yr}, \quad \tau_{\mathrm{Hall}} = \frac{4\pi e n_e L^2}{cB} \sim 10^4 B_{12} \text{ yr}$$

Improved B-field prescription

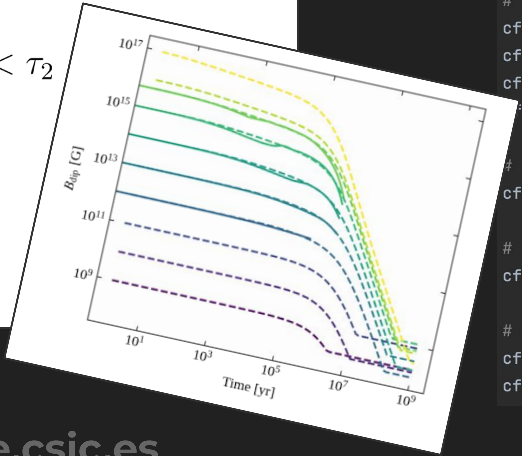
- To provide a more **realistic description of the magnetic field decay**, we **fit an analytical function** to B-field evolution curves obtained from 2D magneto-thermal simulations (Viganó et al., 2021) for various fields. As they are only valid up to 10^6 yr, we extend the **late-time behaviour with a power law** as follows:

$$B(t) = B_0 \left(1+\frac{t}{\tau_1}\right)^{-a_1} \left(1+\frac{t}{\tau_2}\right)^{a_1-a_2} \left(1+\frac{t}{t_{\rm tr}}\right)^{a_2-a_{\rm late}}, \quad \text{if} \quad \tau_1 < \tau_2 < t_{\rm tr}$$

$$B(t) = B_0 \left(1+\frac{t}{\tau_1}\right)^{-a_1} \left(1+\frac{t}{t_{\rm tr}}\right)^{a_2-a_{\rm late}}, \quad \text{if} \quad \tau_1 < t_{\rm tr} < \tau_2$$

$$B(t) = B_0 \left(1+\frac{t}{t_{\rm tr}}\right)^{-a_{\rm late}}, \quad \text{if} \quad t_{\rm tr} < \tau_1 < \tau_2$$

$$\text{with} \quad \tau_1 = A_1 B_0^{-b_1}, \quad \tau_2 = A_2 B_0^{-b_2}$$




```
# Power-law indices.
cfg["a1"]: float = -0.13
cfg["a2"]: float = -3.0

# Timescale parameters, normalizatio
cfg["A1"]: float = 1.0e14
cfg["b1"]: float = -0.8
cfg["A2"]: float = 6.0e8
cfg["b2"]: float = -0.2

# Timescale in [yr] when transitioni
cfg["tau_late"]: float = 2.0e6

# Late time power-law index.
cfg["a_late"]: float = -2.0

# Parameters for a log-normal distri
cfg["B_millisec_mean"] = 8.5
cfg["B_millisec_sigma"] = 0.5
```


graber@ice.csic.es

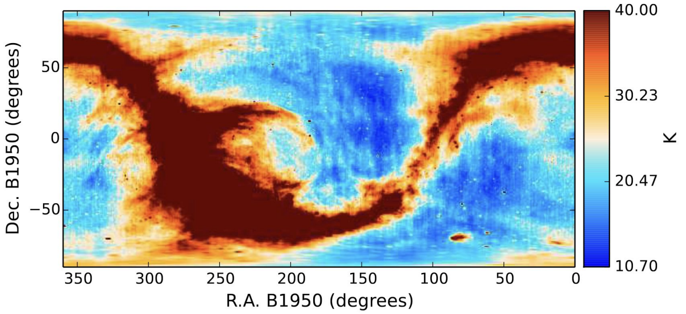
Radiometer equation

$$S_{\text{radio}} = \frac{L_{\text{radio}}}{\Omega_{\text{beam}} d^2}$$

- Following Lorimer & Kramer (2005), we convert the intrinsic (bolometric) radio flux S_{radio} into a flux density measured at a given frequency. We then account for the fact that the intrinsic pulse width is broadened to w_{eff} (due to inter-stellar scattering, dispersion and finite sampling resolution of a detector) and compute the period-averaged flux S_{mean} observed by a radio telescope.

- With these estimates and information specific to the PMPS, SMPS and HTRU surveys, we **determine the signal-to-noise-ratio** for a given simulated pulsar using the **radiometer equation**:

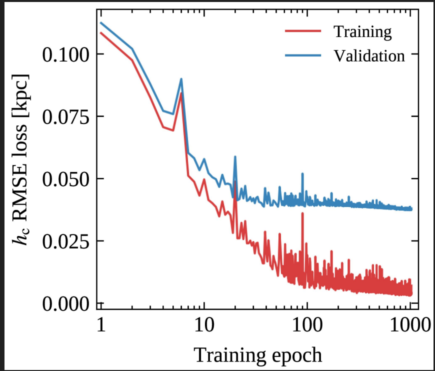
$$S/N = \frac{S_{\text{mean}} G \sqrt{N_{\text{pol}} \Delta\nu \Delta t_{\text{obs}}}}{\beta \left(T_{\text{rec}} + T_{\text{sky}}(l,b)\right)} \sqrt{\frac{P - w_{\text{eff}}}{w_{\text{eff}}}}$$

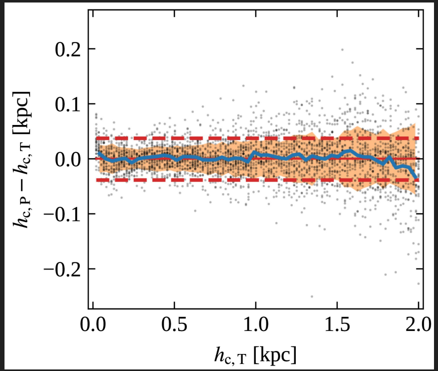


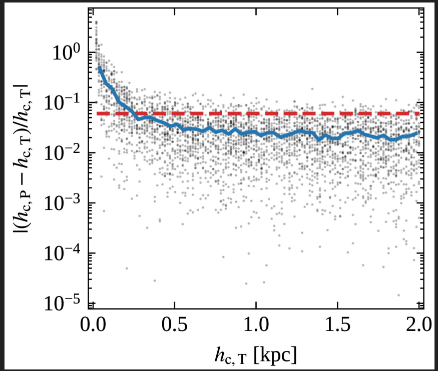
Haslam sky temperature map taken from Remazeilles et al., (2015).

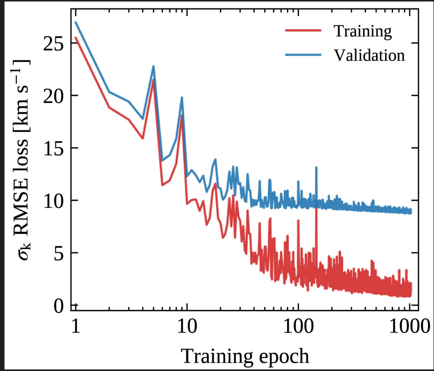
Ronchi et al. (2021) - Training Results

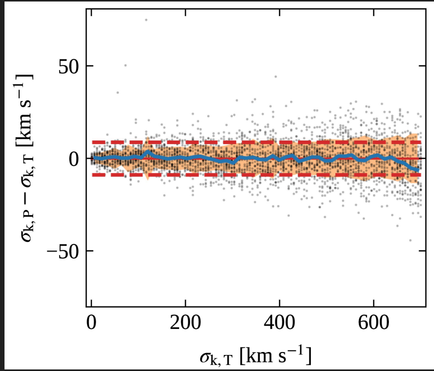
CNN validation results for h_c:
RMSE = 0.038 kpc & MRE = 0.061.

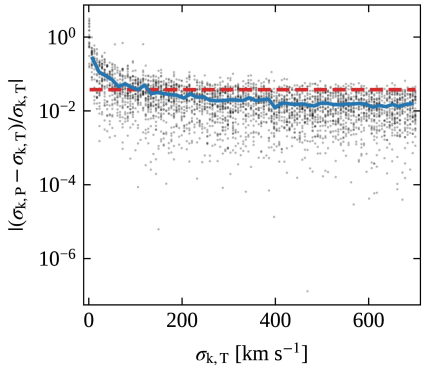










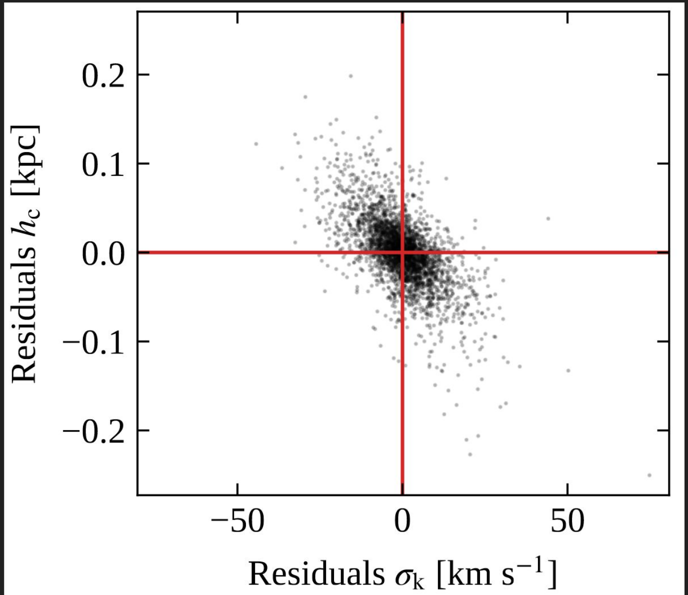


CNN validation results for σ_k:
RMSE = 8.8 km/s & MRE = 0.039.

Ronchi et al. (2021) - Degeneracy

- Histogramming **distances from the Galactic plane** for current mock pulsar populations, we note a **degeneracy between h_c and σ_k**: large scale heights combined with small velocity dispersions lead to same outcomes as small scale heights with large velocity dispersions.

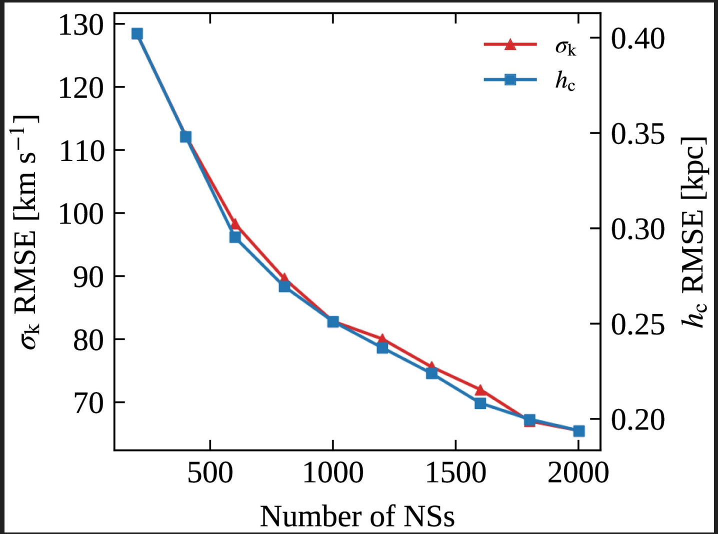
- **CNN recovers the degeneracy!**

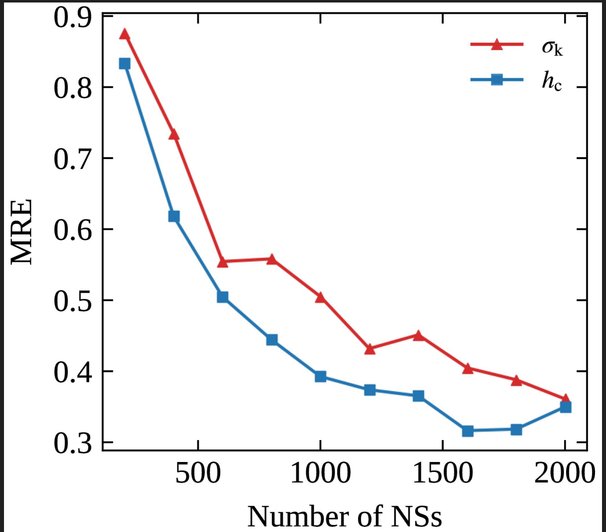


We find an anticorrelation between the residuals in h_c and σ_k.

Ronchi et al. (2021) - Selection Biases

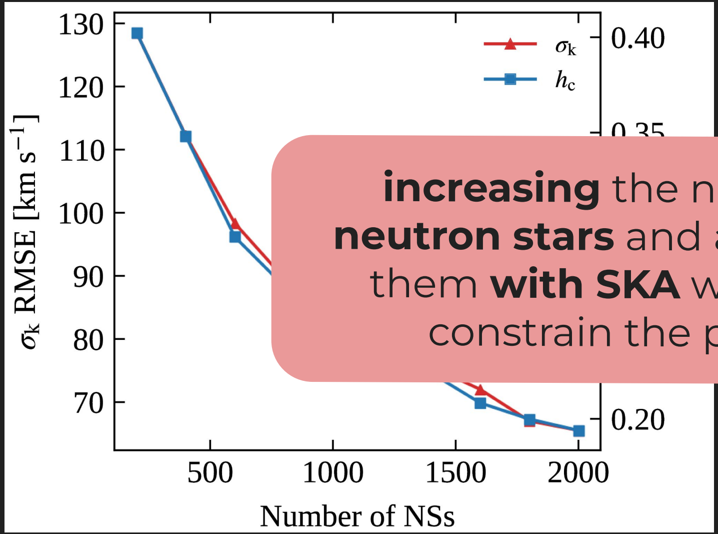
- Analyse the **CNN's predictive power** as a function of available data points (i.e. number of neutron stars) by resampling our fiducial simulation to **incorporate selection biases** from pulsars with **proper-motion measurements**.

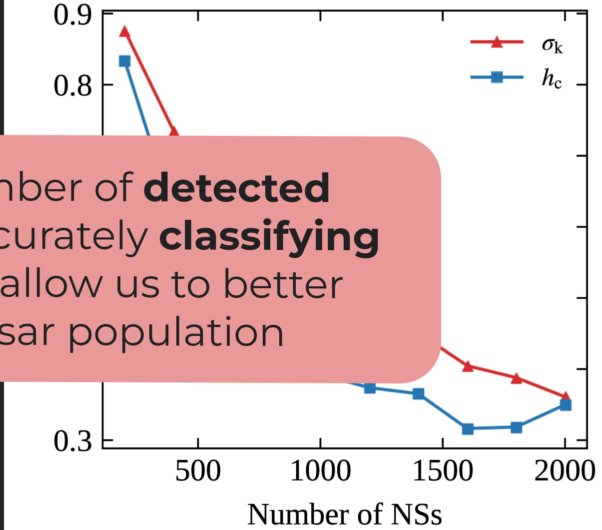




Ronchi et al. (2021) - Selection Biases

- Analyse the **CNN's predictive power** as a function of available data points (i.e. number of neutron stars) by resampling our fiducial simulation to **incorporate selection biases** from pulsars with **proper-motion measurements**.

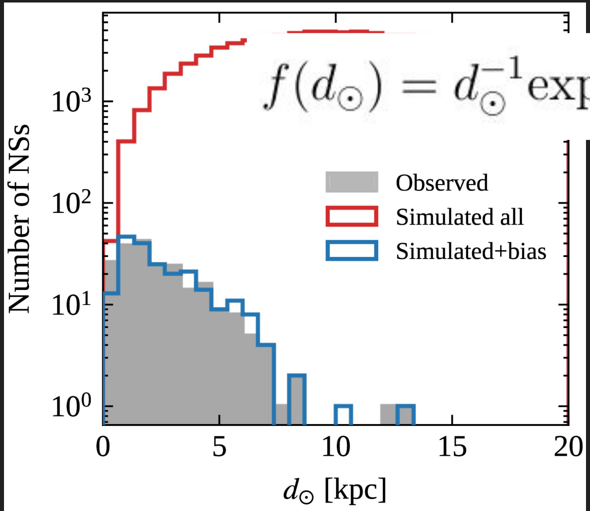




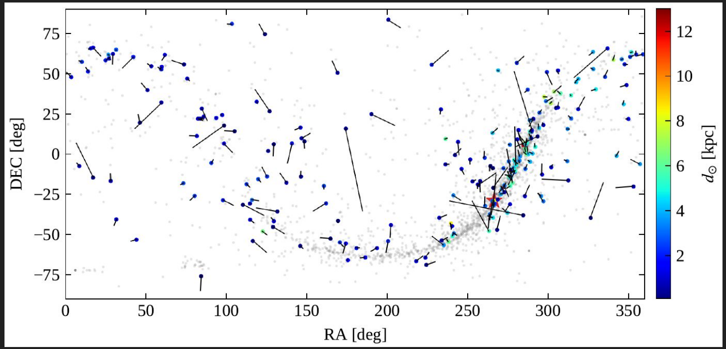
increasing the number of **detected neutron stars** and accurately **classifying** them **with SKA** will allow us to better constrain the pulsar population

Ronchi et al. (2021) - Selection Function

- To incorporate selection effects & observational biases, we use a **phenomenological approach** to reanalyse the CNN performance.



$$f(d_\odot) = d_\odot^{-1} \exp(-d_\odot/2)$$

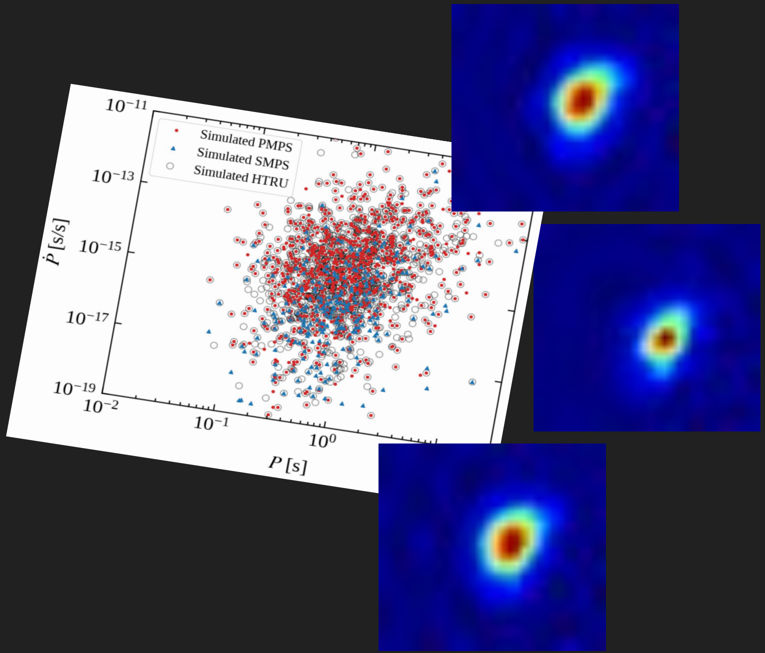


216 isolated NSs with proper motions.

- Use proper motion and distance estimates for 216 isolated pulsars to deduce **empirical selection function** $f(d_\odot)$ and **resample population** with h_c = 0.18 kpc and σ_k = 265 km/s.

Simulated training dataset for sbi

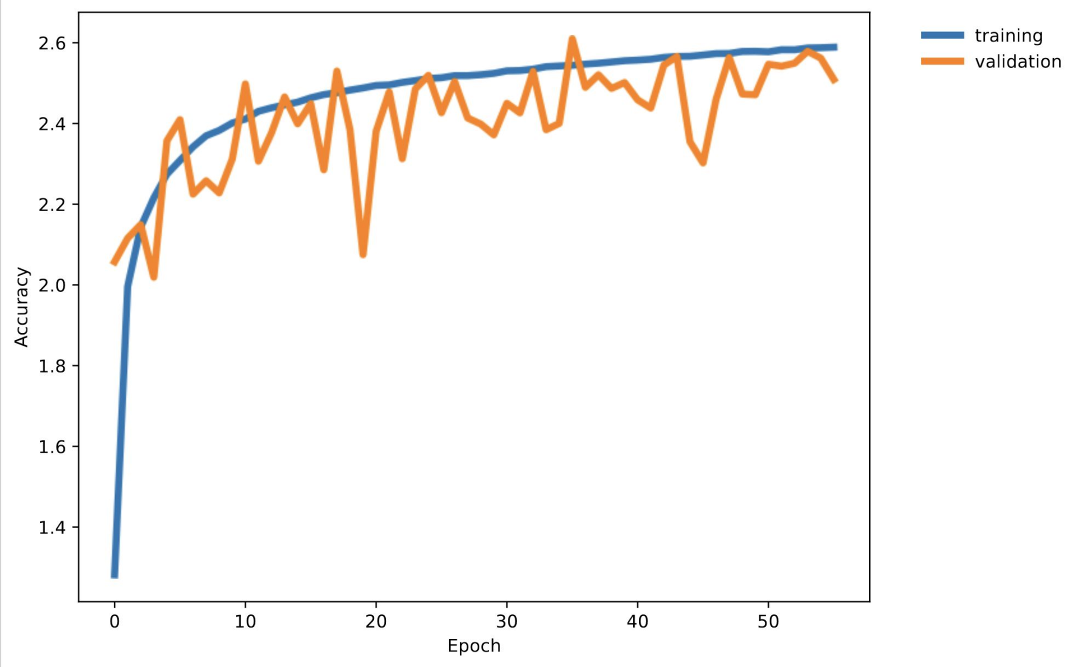
- We vary our **four parameters** for the initial period and magnetic field log-normal distributions as follows:
    - $\mu_P \in$ Uniform(-1, -0.3)
    - $\sigma_P \in$ Uniform(0.1, 0.5)
    - $\mu_B \in$ Uniform(12.0, 14.0)
    - $\sigma_B \in$ Uniform(0.5, 1.0)
- These also correspond to our **flat priors** for the simulation-based inference experiments.



$$\mathcal{P}(P_0) = \frac{\log_{10}(\mathrm{e})}{\sqrt{2\pi}P_0\sigma_P} \exp\left(-\frac{[\log_{10}(P_0) - \mu_P]^2}{2\sigma_P^2}\right)$$

Sbi training behaviour

- **Typical learning behaviour** for our simulation-based inference experiments.
- The **log-probability increases** as a function of training epochs for the training and validation loss.
- We see some variation in the validation loss but **little overfitting**.



Quality checks

- To assess the quality of our neural density estimator, we **perform simulation-based calibration** to assess how well our posterior approximates the true posterior by **plotting the 1D rank statistics** for our four parameters.

- For a well calibrated **unbiased posterior, ranks follow a uniform distribution**: