Uncertainty quantification of transport model results

Zhen Zhang
IFCEN, Sun Yat-Sen University

TMEP
GANIL, Caen
September 15, 2024

Intra-model uncertainties

- **Emulator +Monte-Carlo sampling**

(1)Cozma, arXiv:2407.16411 (dcQMD)

(2)Wang et al., arXiv:2406.07051 (IQMD)

(3)Tsang et al., PLB 853,138661 (2024) (ImQMD-Sky)

(4)Kuttan et al., PRL 131, 202303 (2023) (UrQMD)

(5)Li & Xie, NPA 1039, 122726 (2023) (IBUU)

(6)Morfouace et al., PLB 799, 135045 (2019) (ImQMD)

(7)Margueron's take on Nusym24 (IQMD)

(8)……

- Likelihood (normally Gaussian) :

$$\mathscr{L} \propto \exp(-\frac{\chi^2}{2})$$

$$\chi^2 = \sum_i \frac{\left[y_i(\boldsymbol{\theta}) - y_i^{\exp}\right]^2}{\sigma_i^2}$$

- Prior: $\pi(\boldsymbol{\theta})$

  Usually uniform or Gaussian

- Posterior of $\boldsymbol{\theta}$:

$$p(\theta | \boldsymbol{y}^{\exp}) \propto \mathscr{L}(\boldsymbol{y}^{\exp} | \boldsymbol{\theta}) \pi(\boldsymbol{\theta})$$

Inter-model uncertainty

- **Model comparison** (determine the weighting factor ω_i of each model)
  - **Model selection**: identify the best model with the largest ω_i
  - **Model averaging**: averaging model predictions by
    $$\mathcal{O}_{MA} = \frac{\sum_i \mathcal{O}_i \omega_i}{\sum_i \omega_i}$$

- *Model Selection and Multimodel Inference: A Practical InformationTheoretic Approach* by Burnham & Anderson
- Udo von Toussaint, Rev. Mod. Phys. 83, 943 (2011)

- Frequentist model comparison
  - Akira's information criterion (AIC) :
    AIC $= -2\ln(\mathcal{L}_{\max}) + 2k$
  - Bayesian information criterion (BIC):
    BIC $= -2\ln(\mathcal{L}_{\max}) + k\ln N$
  - $\omega_i = \exp(-\frac{1}{2}\mathrm{AIC})$ or $\omega_i = \exp(-\frac{1}{2}\mathrm{BIC})$
    (only information at the optimal value that maximizes likelihood)

k: number of model parameters

N: number of data points.

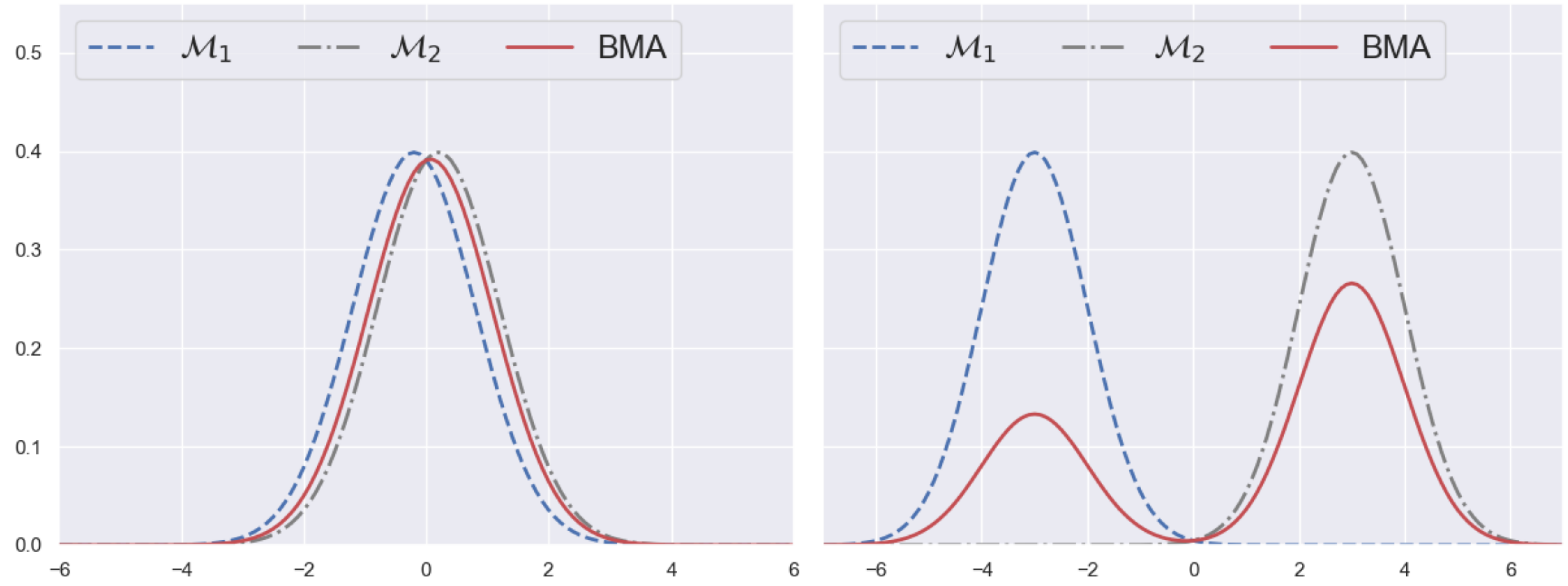
Bayesian model averaging

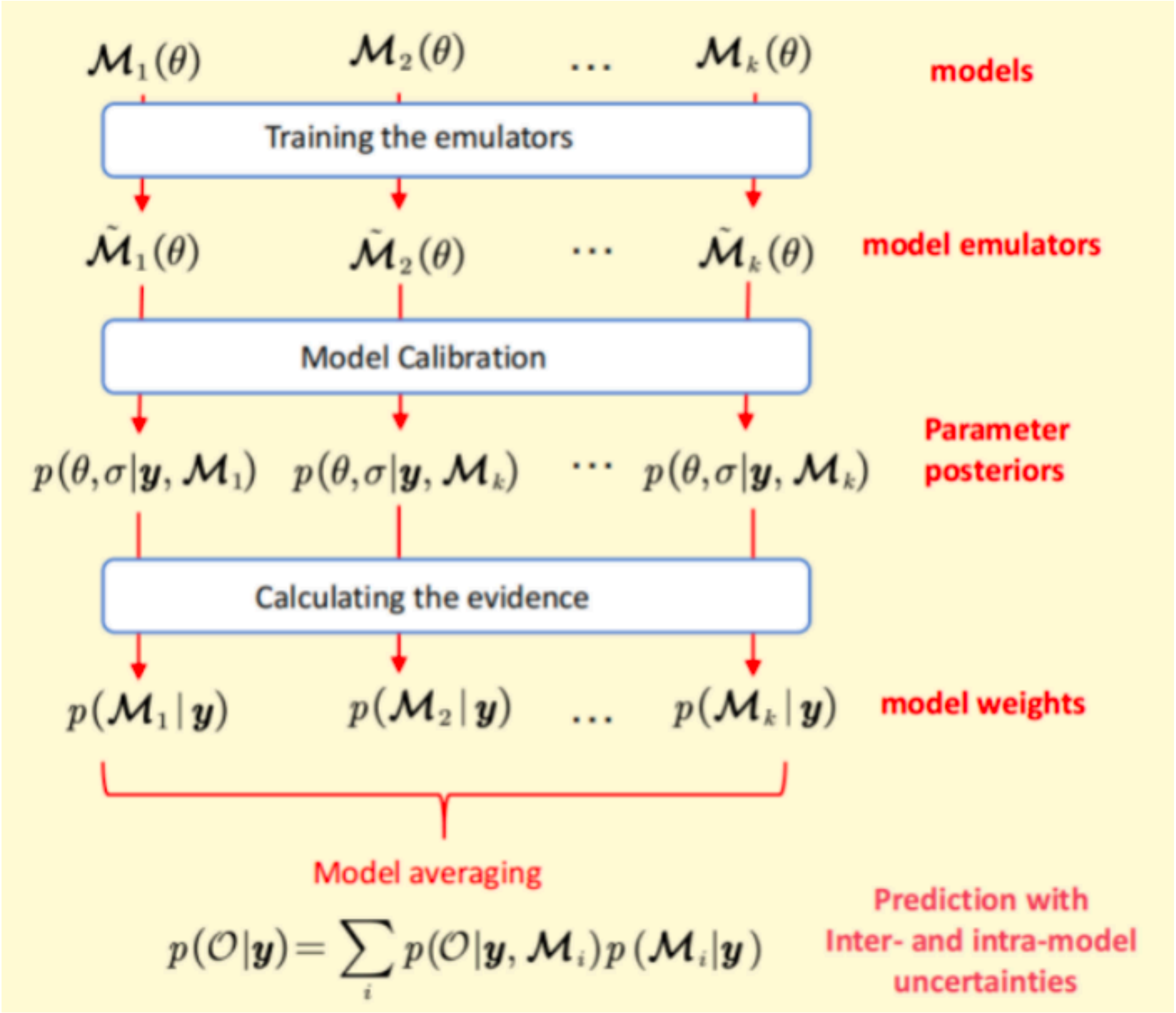
- Weight/probability of each model:

$$\omega_i = p\left(\mathcal{M}_i \mid \mathbf{y}\right) = \frac{p\left(\mathbf{y} \mid \mathcal{M}_i\right)\pi\left(\mathcal{M}_i\right)}{\sum p\left(\mathbf{y} \mid \mathcal{M}_\ell\right)\pi\left(\mathcal{M}_\ell\right)}$$

- **Evidence**:

$$p\left(\mathbf{y} \mid \mathcal{M}_i\right) = \int \mathscr{L}\left(\mathbf{y} \mid \boldsymbol{\theta}, \mathcal{M}_i\right)\pi\left(\boldsymbol{\theta} \mid \mathcal{M}_i\right) d\boldsymbol{\theta}$$





Qiu's talk

A new home work?

- Three parameter K_0 , m^* and η. $(\sigma^*_{NN} = \sigma_{NN}(1 - \eta\rho/\rho_0)$

- Generate 50 (Needs to be tested) training points

- Calculate the selected observables:
  Cl proton $v_1, v_2,$ rapidity distribution… ⁇

- Bayesian inference:

Ashton et al., Nat Rev Methods Primers **2**, 39 (2022)

Code	Methods	Dynamic	Languages	Field	Pub. Year
CosmoNest [60, 61]	ellipsoid	fixed	Fortran	Cosmology	2006
MultiNest [48, 84]	multi-ellipsoid	fixed	Fortran, C/C++, Python	Cosmology	2008
DIAMONDS [249]	multi-ellipsoid	fixed	C++	Astrophysics	2015
nestle [250]	ellipsoid, multi-ellipsoid	fixed	Python	Astrophysics	2015
nessai [90, 91]	normalising flow ellipsoid	fixed	Python	Gravitational waves	2021
(dy)PolyChord [53, 65]	slice	dynamic	Fortran, C/C++, Python	Cosmology	2015
LALInferenceNest [180]	random walk, ensemble, differential evolution	fixed	C	Gravitational waves	2015
Nested_fit [104, 257, 258]	random walk	fixed	Fortran	Atomic physics	2016
cpnest [259]	slice, differential evolution, Gauss, Hamiltonian, ensemble	fixed	Python	Gravitational waves	2017
pymatnest [44]	random walk, Galilean, symplectic Hamiltonian	fixed	Python	Materials	2017
NNest [261]	normalising flow random walk	fixed	Python	Cosmology	2019
DNest5 [55]	user-defined, random walk	diffusive	C++	Astrophysics	2020
BayesicFitting [263]	random walk, slice, Galilean, Gibbs	fixed	Python	Astronomy	2021
dynesty [52]	ellipsoid, multi-ellipsoid, MLFriends & Gauss, slice, Hamiltonian	dynamic	Python	Astrophysics	2020
UltraNest [92]	MLFriends + ellipsoid & Gauss, hit-and-run, slice	reactive	Python, Julia, R, C/C++, Fortran	Astrophysics	2020
jaxns [266]	multi-ellipsoid & slice	fixed	jax	Astronomy	2021

Table 2 | **Comparison of NS codes.** The first two groups are region samplers and step samplers, respectively, whereas the third group offers both. Dynamic implementations allow the number of live points to be changed during a run. We show the language in which the NS code was written followed by any additional languages for which interfaces exist, and the field from which the code originated (though most are general purpose codes).