

**Title:** mochi\_class: A tool to streamline cosmological analyses of Horndeski's gravity

**Speakers:** Matteo Cataneo

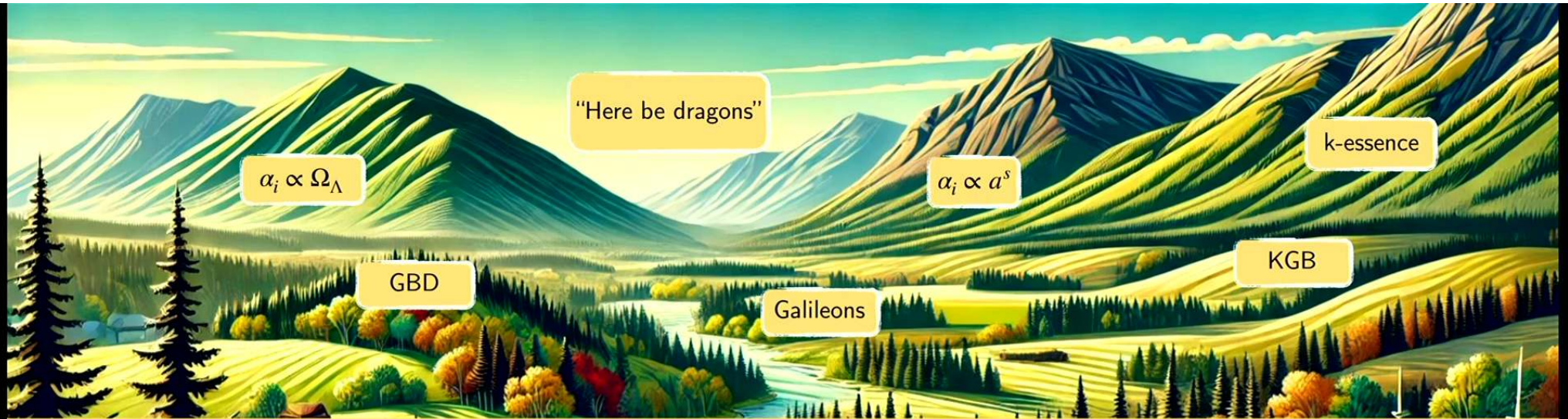
**Collection/Series:** 50 Years of Horndeski Gravity: Exploring Modified Gravity

**Date:** July 17, 2024 - 2:30 PM

**URL:** <https://pirsa.org/24070068>

**Abstract:**

I will introduce mochi\_class, a refined version of the popular Einstein-Boltzmann solver hi\_class optimised for calculations within Horndeski's gravity framework. Thanks to (i) a re-parametrisation of Horndeski functions, (ii) a numerically stable quasi-static approximation, and (iii) support for time-dependent inputs, mochi\_class enhances hi\_class capabilities and nicely complements other public Einstein-Boltzmann solvers. Additionally, I will present a non-parametric approach that, when integrated with Principal Component Analysis, can effectively reconstruct Horndeski functions from well-studied modified gravity models, extending the exploration of scalar-tensor theories beyond conventional parametrisations. I will conclude by highlighting practical applications where mochi\_class can prove instrumental in analysing current and forthcoming large-scale structure data.



mochi\_class:

## A numerical tool to streamline cosmological analyses of Horndeski gravity

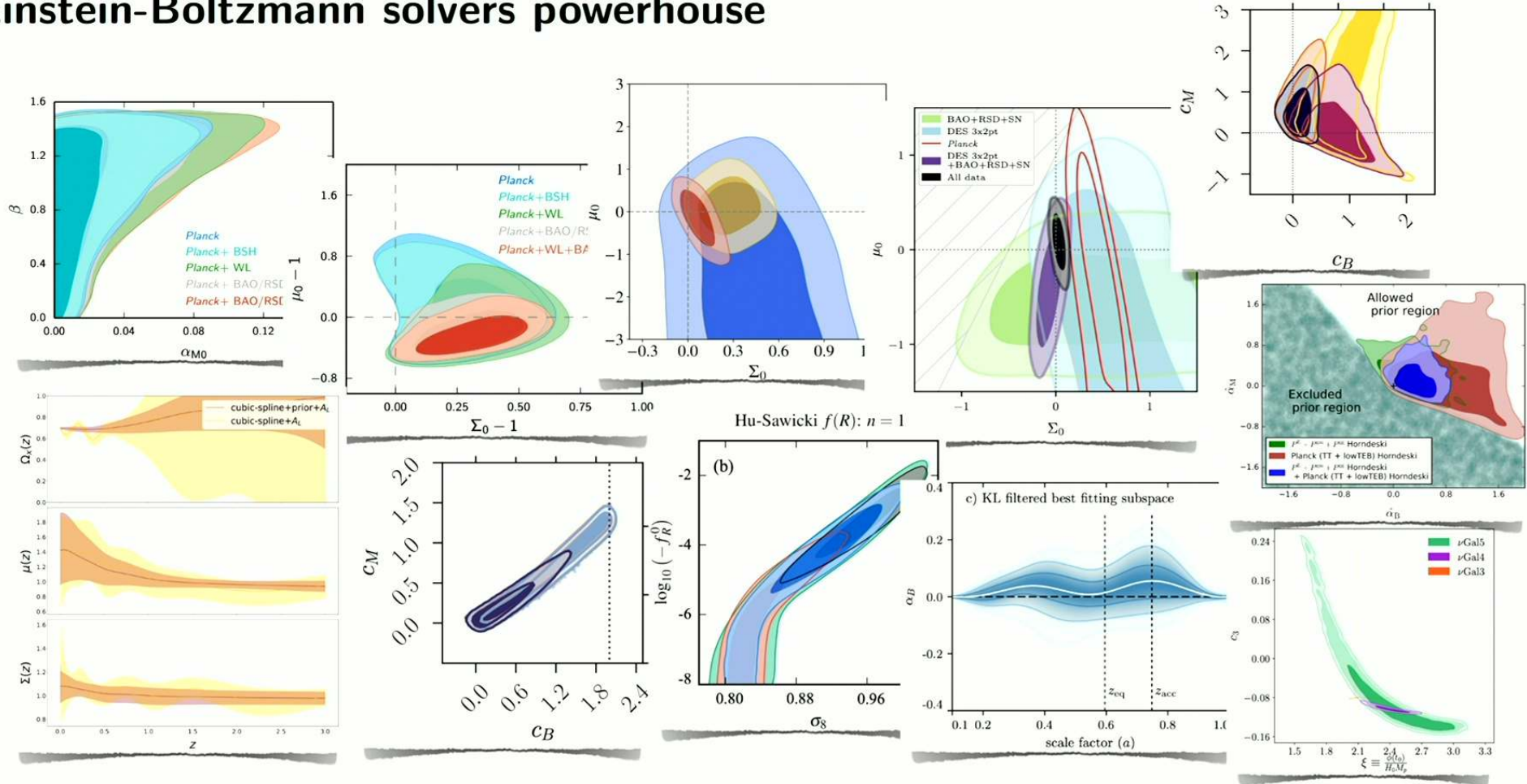
*arXiv:2407.11968*

Matteo Cataneo, Perimeter Institute, 17/07/2024

In collaboration with: Emilio Bellini (SISSA)




# Einstein-Boltzmann solvers powerhouse




Planck 2015, DES 2019, DES 2023, Seraille et al 2024, Noeller & Nicola 2018, Wang et al. 2023, Hu et al. 2016, Raveri 2019, Renk et al. 2017, Spurio Mancini et al. 2019...And many, many more!

# The Einstein-Boltzmann solvers triad of marvels




MGCAMB

- ✓ QSA
- ✓ Specific models
- ✓ Splined  $\{\mu, \Sigma\}$
- ✓  $\{\mu, \Sigma\}, \{\mu, \gamma\}, \{Q, R\}$  parametrisations



EFTCAMB

- ✓ EFT of DE
- ✓ No QSA
- ✓ Specific models
- ✓ Lagrangian-to-EFT mapping
- ✓ EFT functions and  $\alpha_i$ 's parametrisations



hi\_class

- ✓ EFT of Horndeski w/  $\alpha_i$ 's
- ✓ Selective QSA
- ✓ Specific models
- ✓ Lagrangian-to-EFT mapping
- ✓  $\alpha_i$ 's parametrisations

But see also ISiTGR, COOP, mgclass...

## Already got all we need, or do we?

	Horndeski gravity	EFT mapping	General parametrisation	Fully dynamical evolution	QSA	Maths stability	Python wrapper
hi_class	✓	(✓)(1)	✗	✓	(✓)(2)	✗	✓
EFTCAMB	✓	(✓)(1)	✗	✓(3)	✗	✓	✗
MGCAMB	(✓)(4)	✗	(✓)(5)	✗	✓	N/A	✓

(1) can lead to numerical instabilities

(2) can be inaccurate

(3) to speed up computation users can adjust two parameters, EFTtoGR and EFTTurnOnPiInitial, accuracy might be affected

(4) only a few particular models in the Horndeski class

(5) spline of phenomenological  $\{\mu, \Sigma\}$

**GAP: none of the public codes has all desired features**

# mochi\_class: MODELLING OPTIMISATION TO COMPUTE HORNDESKI IN CLASS

	Horndeski gravity	EFT mapping	General parametrisation	Fully dynamical evolution	QSA	Maths stability	Python wrapper
mochi_class	✓	✓	✓	✓	✓	✓	✓

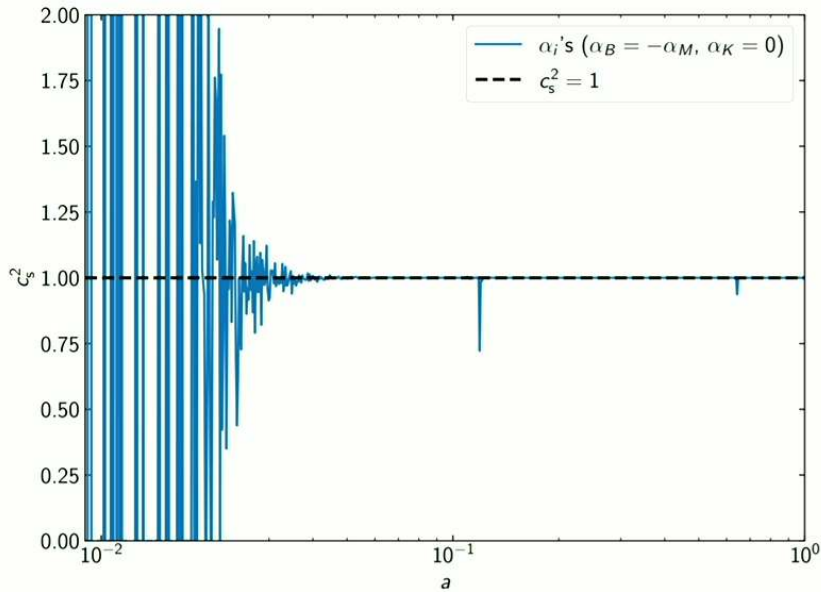
Restricted to theories with  $\alpha_T = 0$ :

$$\mathcal{L}_{\text{SH}} = K(\phi, X) + G_4(\phi)R - G_3(\phi, X)\square\phi$$



## mochi\_class: MODELLING OPTIMISATION TO COMPUTE HORNDESKI IN CLASS

- 1) Implements manifestly stable parametrisation (Kennedy et al. 2018) & 2) Avoids numerical instabilities in  $c_s^2$



Stability conditions:

$$M_*^2 > 0$$

$$D_{\text{kin}} = \alpha_K + 3\alpha_B^2/2 > 0$$

$$c_s^2 = \frac{1}{D_{\text{kin}}} \left[ (2 - \alpha_B) \left( -\frac{H'}{aH^2} + \frac{1}{2}\alpha_B + \alpha_M \right) - \frac{3(\rho_m + p_m)}{H^2 M_*^2} + \frac{\alpha'_B}{aH} \right] > 0$$

Solution:  $\alpha_K, \alpha_B, \alpha_M \implies M_*^2, D_{\text{kin}}, c_s^2$

## mochi\_class: MODELLING OPTIMISATION TO COMPUTE HORNDESKI IN CLASS

3) Takes general functions of time as input

```
else if(pba->gravity_model_smg == nkgb){  
  
    /* Action is  
  
    -X + 1/n * g^{(2n-1)/2} Lambda (X/Lambda^4)^n box(phi)  
  
    g was picked like this so that it approx. remains g*Omega_smg0 ~ 0(1) for all n  
    Note that for n=1/4 the Lambda mass scales cancels out, so we set it to 1.  
    */  
  
    double g = pba->parameters_smg[0];  
    double npow = pba->parameters_smg[1];  
    double ngpow = copysign(1.,g)*pow(fabs(g), (2.*npow-1.)/2.)/npow;  
    double H0=pba->H0;  
  
    pgf->G2 = -X;  
    pgf->G2_X = -1.;  
  
    // pgf->G3 = 1/n g^{(2n-1)/2} Lambda (X/Lambda^4)^n  
  
    pgf->G3_X = npow*ngpow*pow(X, npow-1)/pow(H0, 2*npow);  
    pgf->G3_XX = npow*(npow-1)*ngpow*pow(X, npow-2)/pow(H0, 2*npow);  
}
```

```
if (pba->gravity_model_smg == propto_omega) {  
  
    double c_k = pba->parameters_2_smg[0];  
    double c_b = pba->parameters_2_smg[1];  
    double c_m = pba->parameters_2_smg[2];  
    double c_t = pba->parameters_2_smg[3];  
  
    pvecback[pba->index_bg_kineticity_smg] = c_k*Omega_smg;  
    pvecback[pba->index_bg_braiding_smg] = c_b*Omega_smg;  
    pvecback[pba->index_bg_tensor_excess_smg] = c_t*Omega_smg;  
    pvecback[pba->index_bg_mpl_running_smg] = c_m*Omega_smg;  
    pvecback[pba->index_bg_delta_M2_smg] = delta_M_pl; //M2-1  
    pvecback[pba->index_bg_M2_smg] = 1.+delta_M_pl;  
}
```

Solution:  $M_*^2, D_{\text{kin}}, c_s^2, \tilde{\rho}_\phi$  or  $w_\phi$  can be read from file or input as arrays

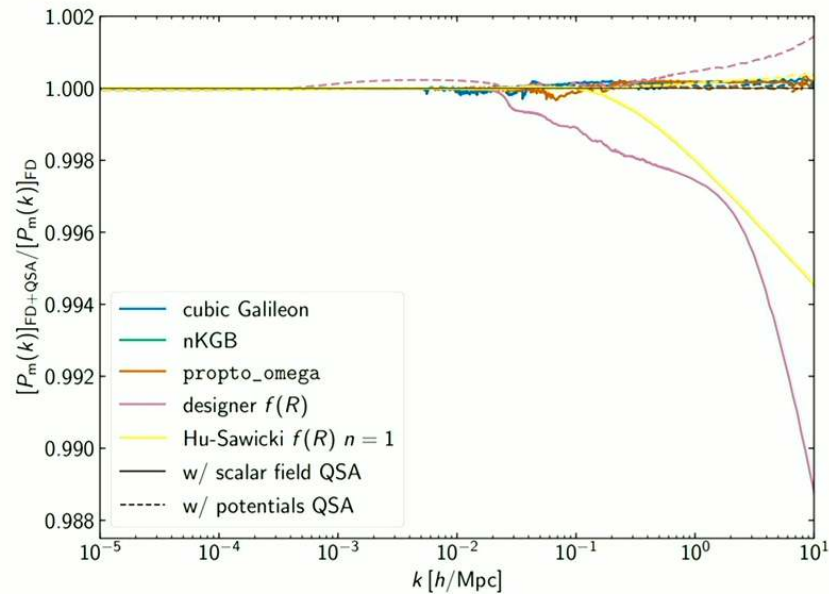
## mochi\_class: MODELLING OPTIMISATION TO COMPUTE HORNDESKI IN CLASS

- 4) Replaces hi\_class QSA with  $\{\mu, \gamma\}$  à la MGCAMB, and connects it to the EFT functions (Pogosian & Silvestri 2016)

In hi\_class:

$$V_X \approx \frac{\mathcal{S}(\tau, k, \eta, \delta\rho, \delta p)}{a^2 H^2 \mu_{\text{eff}}^2(\tau, k)}$$

$$V'_X \approx \frac{d}{d\tau} \left[ \frac{\mathcal{S}(\tau, k, \eta, \delta\rho, \delta p)}{a^2 H^2 \mu_{\text{eff}}^2(\tau, k)} \right]$$

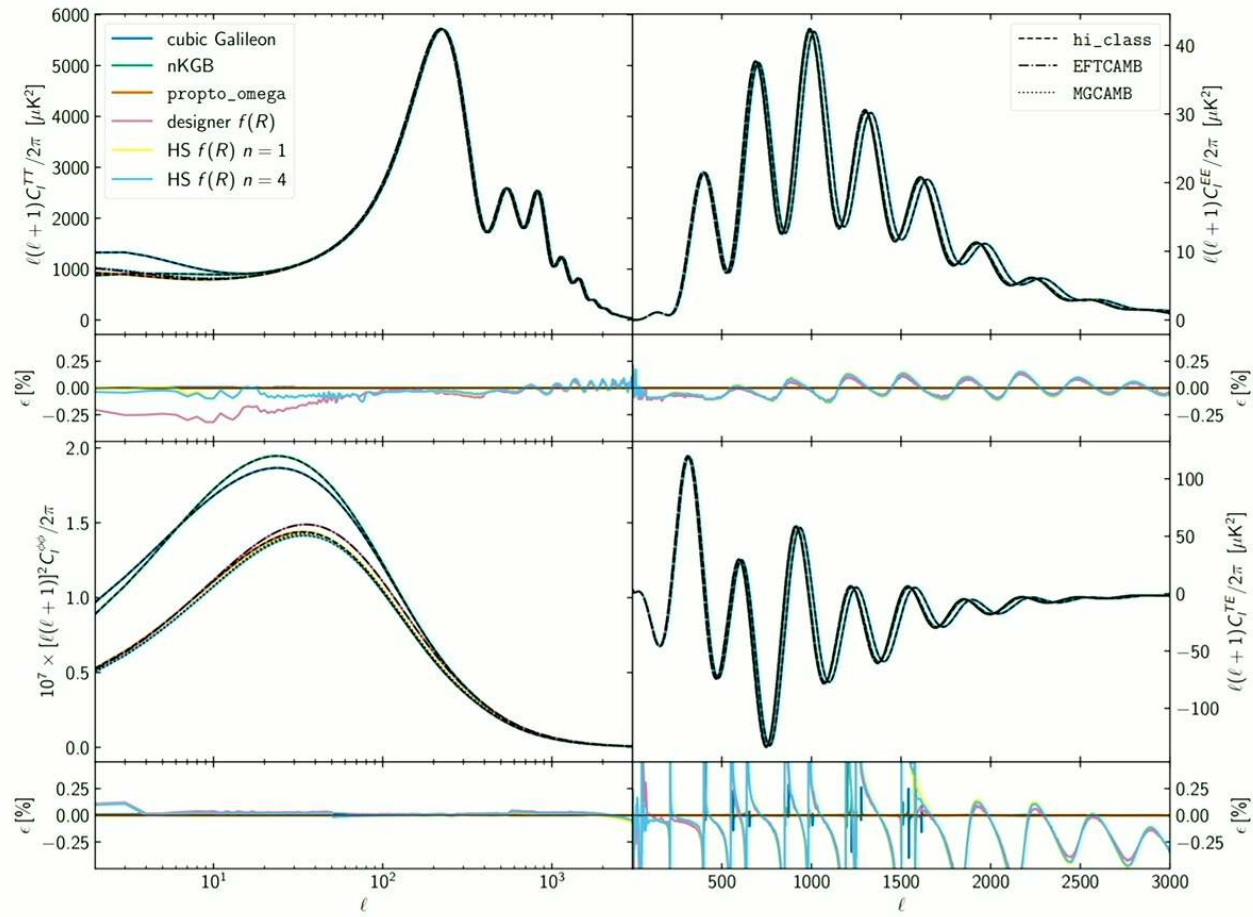


Solution:

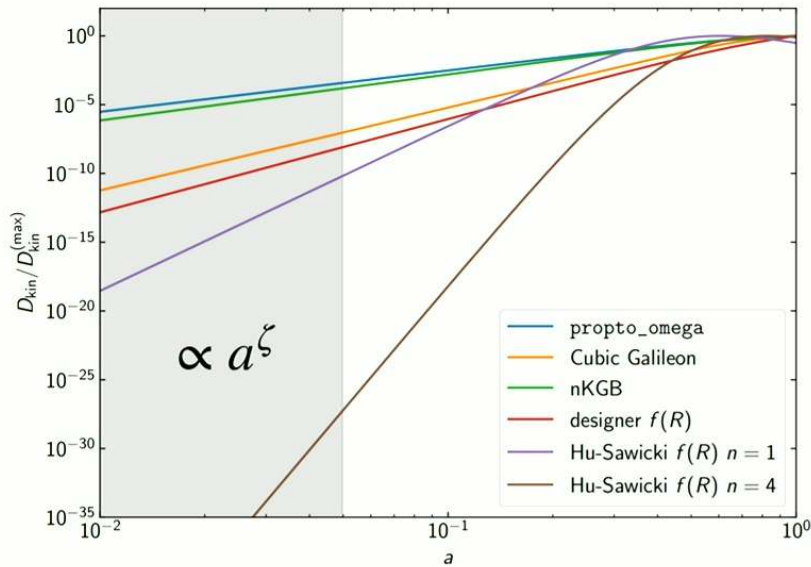
$$k^2 \Psi = -4\pi G a^2 \mu(a, k) \rho_m \Delta_m$$

$$\gamma(a, k) = \Phi / \Psi$$

# Code comparison (1)



# Extending common parametrisations



$$\Delta M_*^2 \approx A_M e^{\zeta_M x + \Delta_1(x)}$$

$$c_s^2 \approx (A_{c_s} e^{\zeta_{c_s} x} + C_{c_s}) e^{\Delta_3(x)}$$

$$D_{\text{kin}} \approx A_D e^{\zeta_D x + \Delta_2(x)}$$

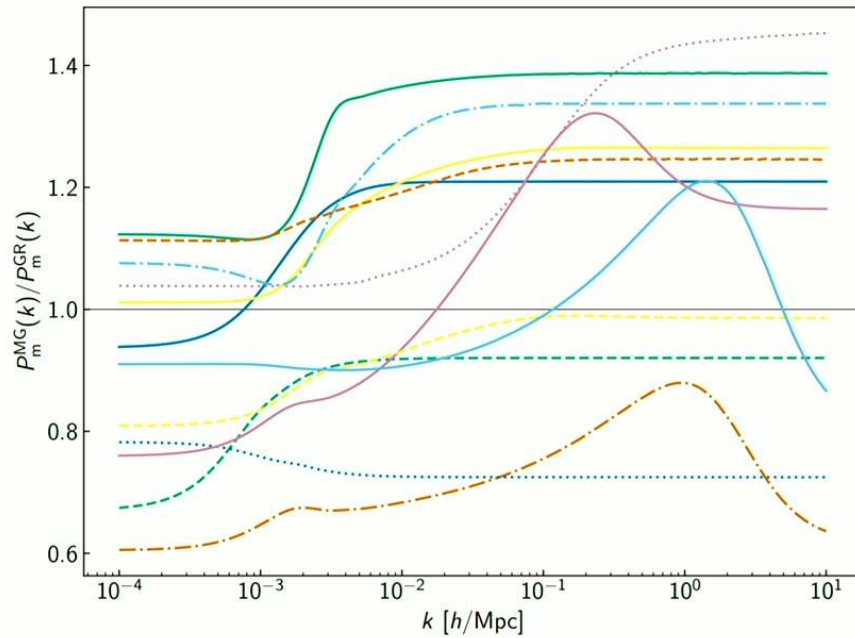
$$\tilde{\rho}_\phi \approx (A_\rho e^{\zeta_\rho x} + C_\rho) \Delta_4(x)$$

And we want

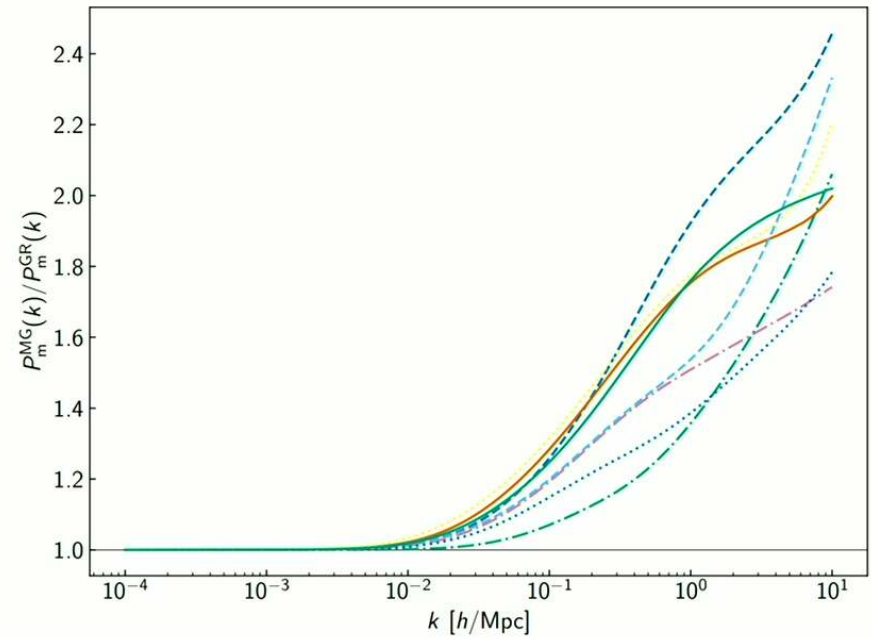
- 1)  $\Delta_{1-3}(x) \longrightarrow 0, \Delta_4(x) \longrightarrow 1$  for  $x \ll 0$
- 2) All  $\Delta_i(x)$  to be varying smoothly

# Let's generate new models!

Scalar Horndeski (31 parameters) - 5% 'survival rate'



$f(R)$ -like (16 parameters) - 20% 'survival rate'





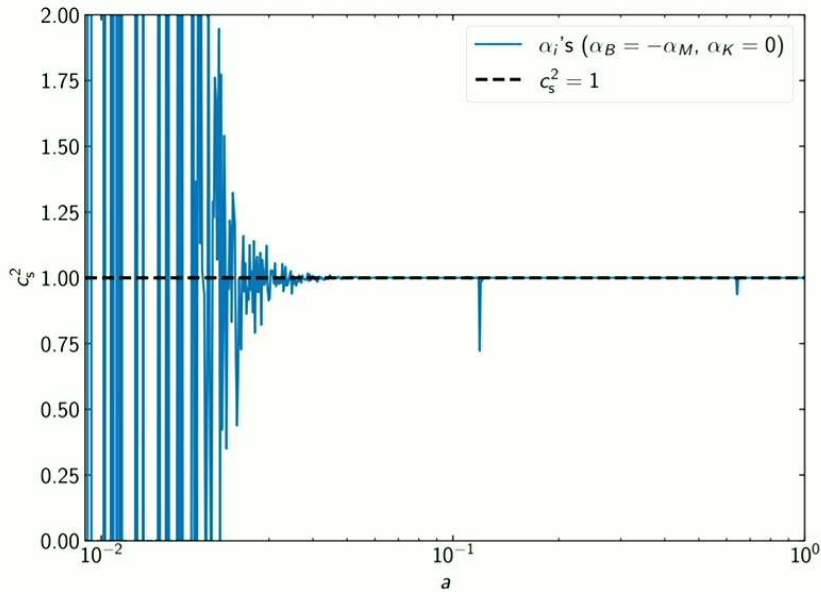
mochi\_class



[github.com/mcataneo/mochi\\_class\\_public](https://github.com/mcataneo/mochi_class_public)

## mochi\_class: MODELLING OPTIMISATION TO COMPUTE HORNDESKI IN CLASS

- 1) Implements manifestly stable parametrisation (Kennedy et al. 2018) & 2) Avoids numerical instabilities in  $c_s^2$



Stability conditions:

$$M_*^2 > 0$$

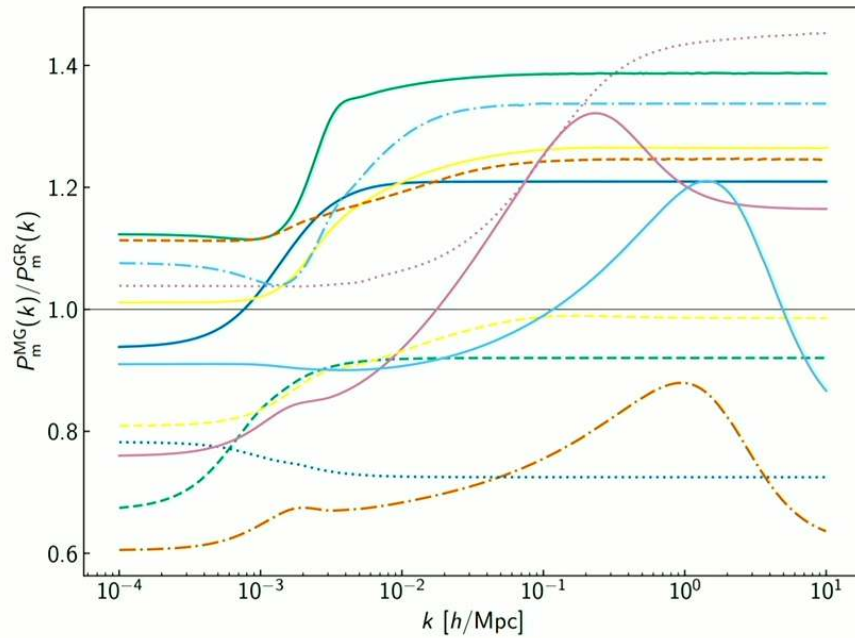
$$D_{\text{kin}} = \alpha_K + 3\alpha_B^2/2 > 0$$

$$c_s^2 = \frac{1}{D_{\text{kin}}} \left[ (2 - \alpha_B) \left( -\frac{H'}{aH^2} + \frac{1}{2}\alpha_B + \alpha_M \right) - \frac{3(\rho_m + p_m)}{H^2 M_*^2} + \frac{\alpha'_B}{aH} \right] > 0$$

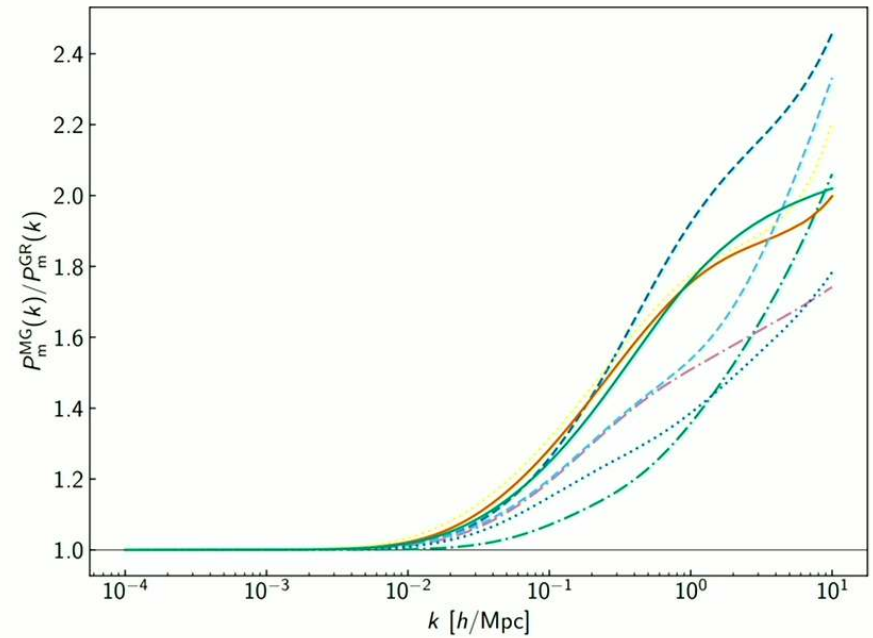
Solution:  $\alpha_K, \alpha_B, \alpha_M \implies M_*^2, D_{\text{kin}}, c_s^2$

# Let's generate new models!

Scalar Horndeski (31 parameters) - 5% 'survival rate'



$f(R)$ -like (16 parameters) - 20% 'survival rate'



## Mathematical stability

- Scalar field fluctuations EOM

$$\mathcal{A}(\tau)V_X'' + \mathcal{B}(\tau)V_X' + [\mathcal{C}(\tau) + k^2\mathcal{D}(\tau)] V_X = \mathcal{E}(\tau, k)$$

- If  $\mathcal{B}^2(\tau) - 4\mathcal{A}(\tau)[\mathcal{C}(\tau) + k^2\mathcal{D}(\tau)] > 0$ :

$$\frac{-\mathcal{B}(\tau) \pm \sqrt{\mathcal{B}^2(\tau) - 4\mathcal{A}(\tau)[\mathcal{C}(\tau) + k^2\mathcal{D}(\tau)]}}{2\mathcal{A}(\tau)} < \xi H_0$$

- If  $\mathcal{B}^2(\tau) - 4\mathcal{A}(\tau)[\mathcal{C}(\tau) + k^2\mathcal{D}(\tau)] < 0$ :

$$-\frac{\mathcal{B}(\tau)}{2\mathcal{A}(\tau)} < \xi H_0$$