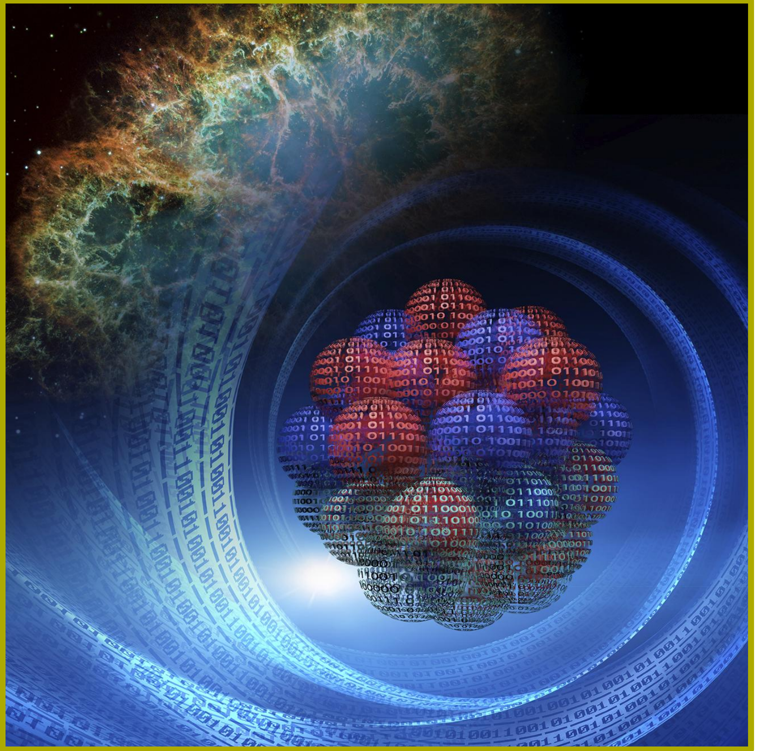
WOUTER RYSSENS



Andy Sproles, ORNL

A new generation of Skyrme ~~mass~~ models on a mesh

Wouter Ryssens

22nd of November 2023

wryssens.com

wryssens@ulb.be

Table of contents

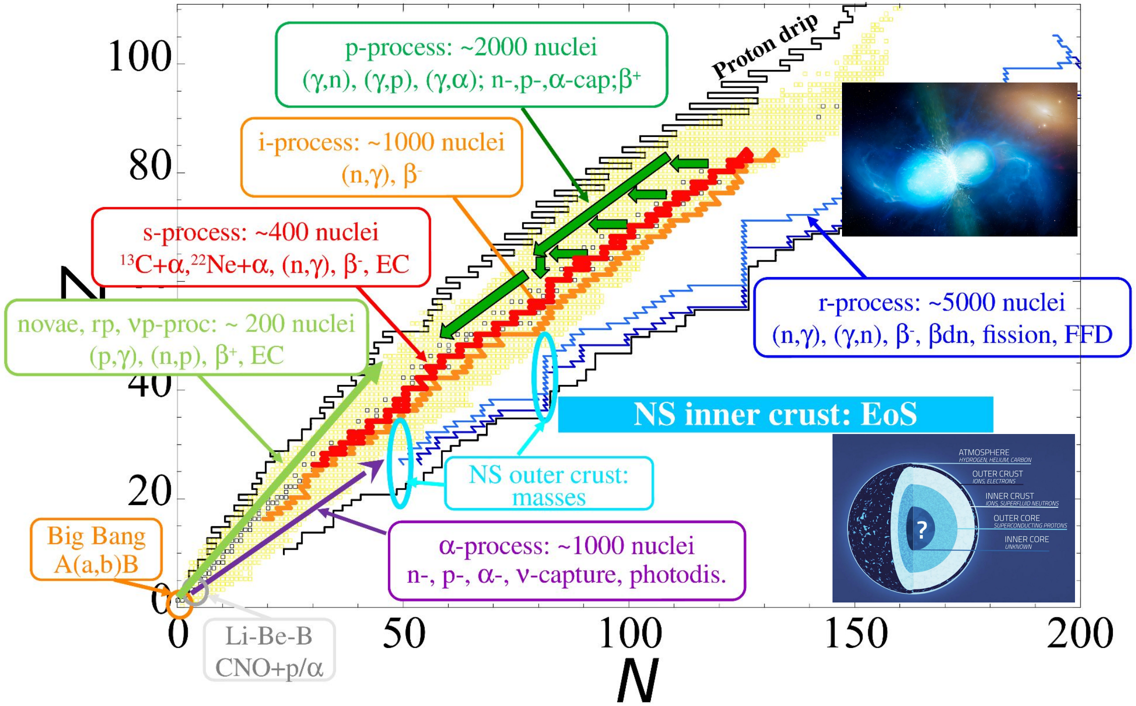
The Brussels state of mind

Extrapolations in
- nucleon number
- energy
- temperature
- density
- ……

and all of that for
- **~7000** nuclei
- **many** reactions

what we need is models that should be

1. **predictive**….
2. but also **complete**

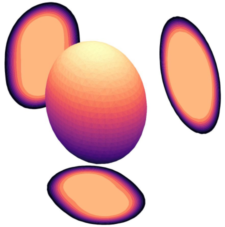


Brussels-**Sk**yrme-on-a-**G**rid: BSkG

BSkG1: G. Scamps et al., EPJA **57**, 333 (2021).
BSkG2: W. Ryssens et al., EPJA **58**, 246 (2022).
W. Ryssens et al., EPJA **59**, 96 (2023).
BSkG3: G. Grams et al., EPJA 59, 270 (2023).

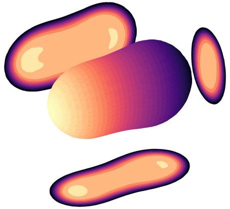
BSkG1 (2021)

- fitted to 2457 masses
- fitted to 884 charge radii
- includes triaxial deformation



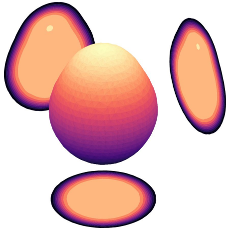
BSkG2 (2022)

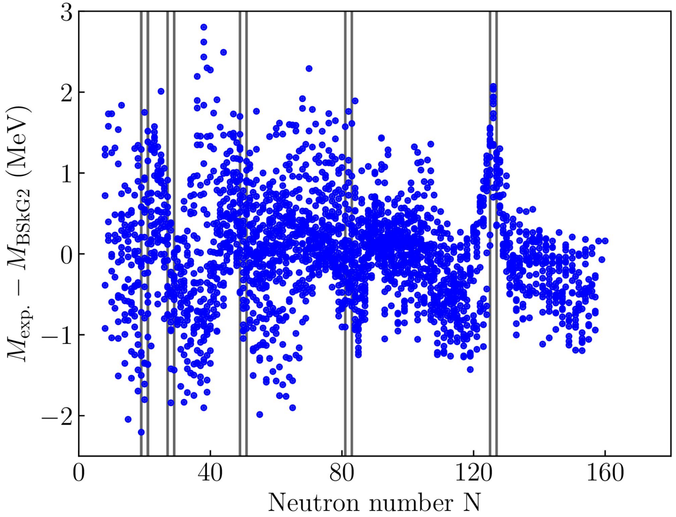
- fitted to 45 fission barriers
- includes spins, currents,...



BSkG3 (2023)

- larger max. neutron star mass
- includes octupole deformation





Rms σ	BSkG1	BSkG2	BSkG3
Masses [MeV]	0.741	0.678	0.631
Radii [fm]	0.024	0.027	0.024
Prim. barriers [MeV]	0.88	0.44	0.33
Secon. barriers [MeV]	0.87	0.47	0.51
Fission isomers [MeV]	1.0	0.49	0.34
Max. NS mass [$M_\odot$]	1.8	1.8	2.3

Ingredients: BSkG3

$$
\begin{aligned}
E_{\text{tot}} = {} & E_{\text{kin}} \\
& + E_{\text{Skyrme}} \\
& + E_{\text{pairing}} \\
& + E_{\text{Coul}} \\
& + E_{\text{CM}}^{(1)} \\
& + E_{\text{CM}}^{(2)} \\
& + E_{\text{rot}} \\
& + E_{\text{vib}} \\
& + E_{\text{Wigner}} \,.
\end{aligned}
$$

Variationally treated

Skyrme form with additional density dependencies and well-defined time-odd terms (**17** parameters)

Mimicking pairing in INM + gradient terms (**3**)

includes finite-size effects of protons and neutrons

Semi-variationally treated

P. Da Costa et al., arXiv:2310.05090

crucial to include for deformation properties

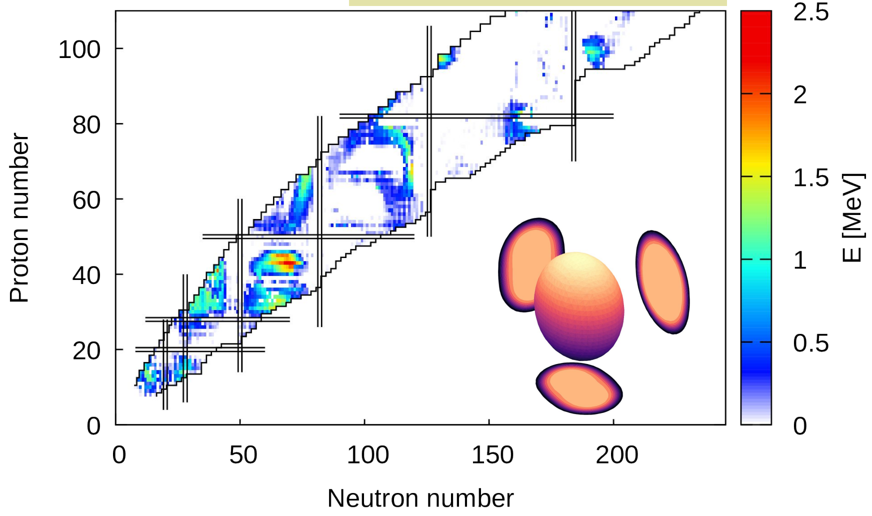
based on simple cranking rotational model (**3**)

simple rescaling of rotational correction (**2**)

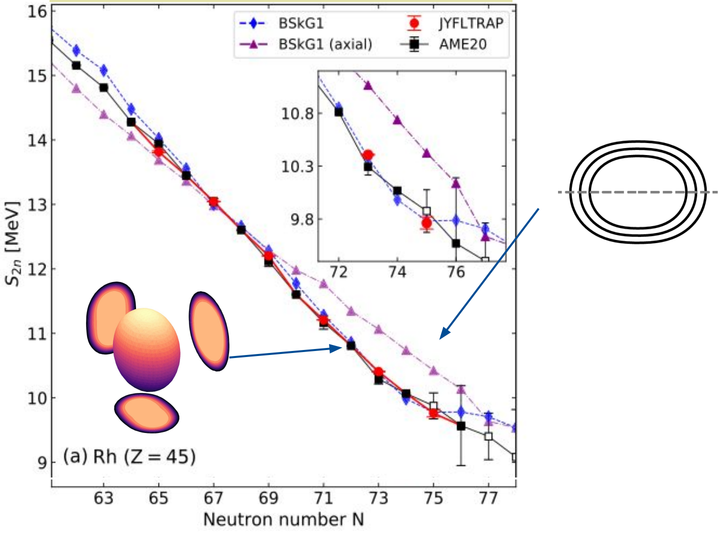
simple formula; mostly active for light N~Z (**4**)

Successes: masses

G. Scamps et al., EPJA **57**, 333 (2021).



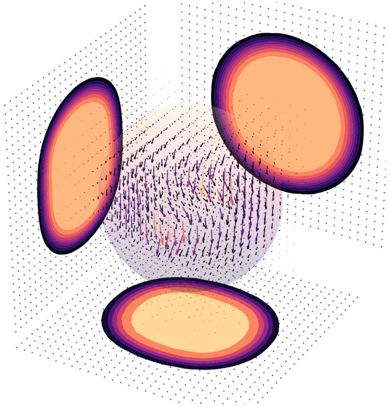
M. Hukkanen, W.R. et al., PRC **107**, 014306 (2023).

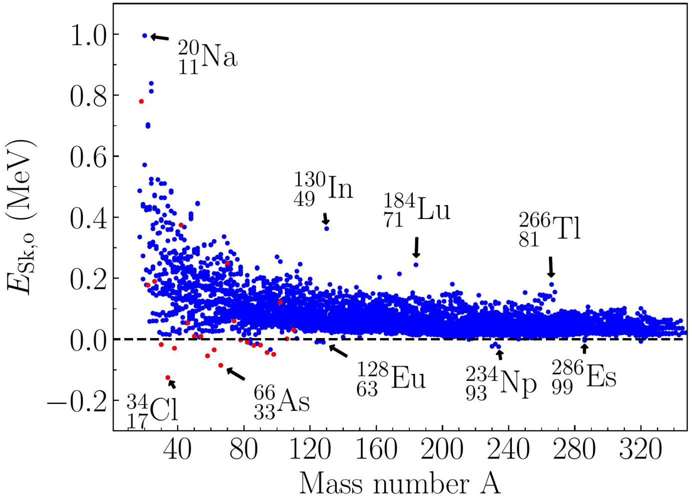


Triaxial deformation

- many nuclei are affected
- effects up to 2.5 MeV near Z~44
- does help reproduce trends, e.g. Rh

Successes: masses

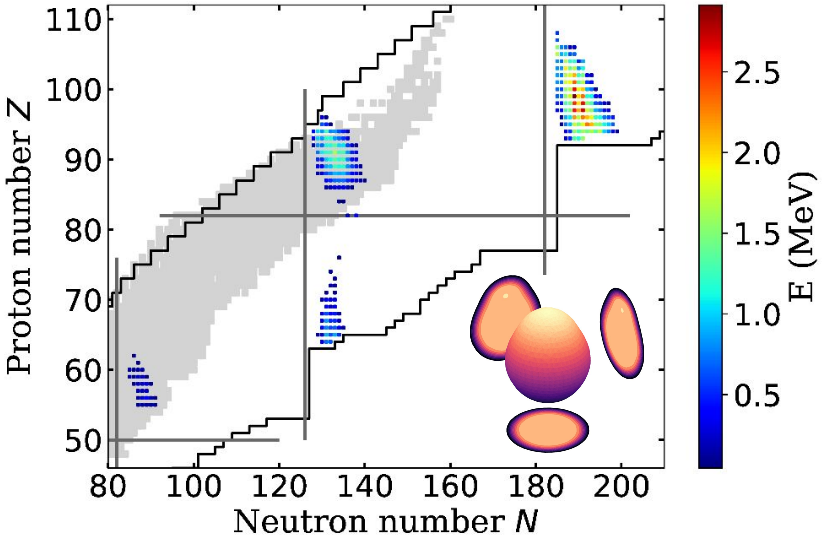


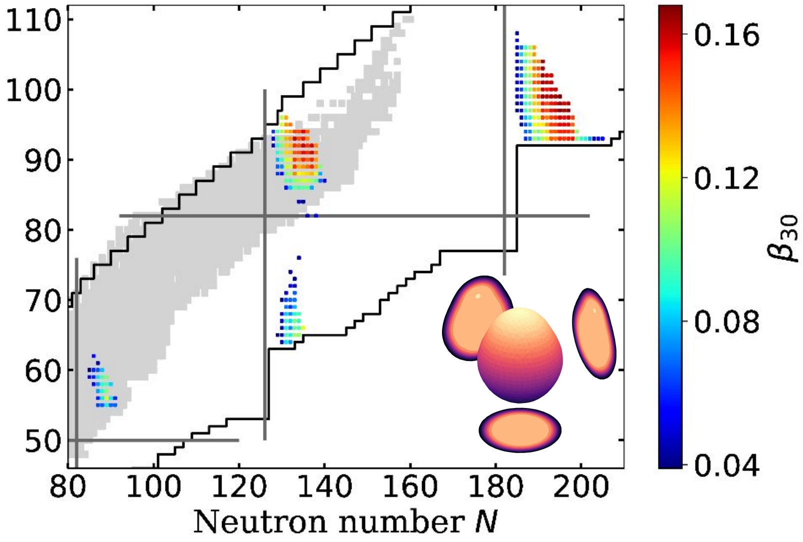


Time-odd terms

- small impact on the masses
- globally repulsive
- first time checked on this scale!
- first step towards other observables

Successes: masses



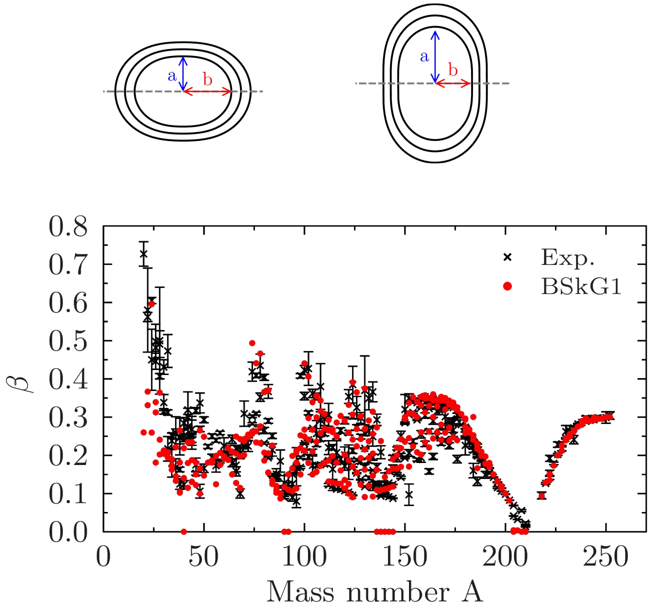


Reflection asymmetry

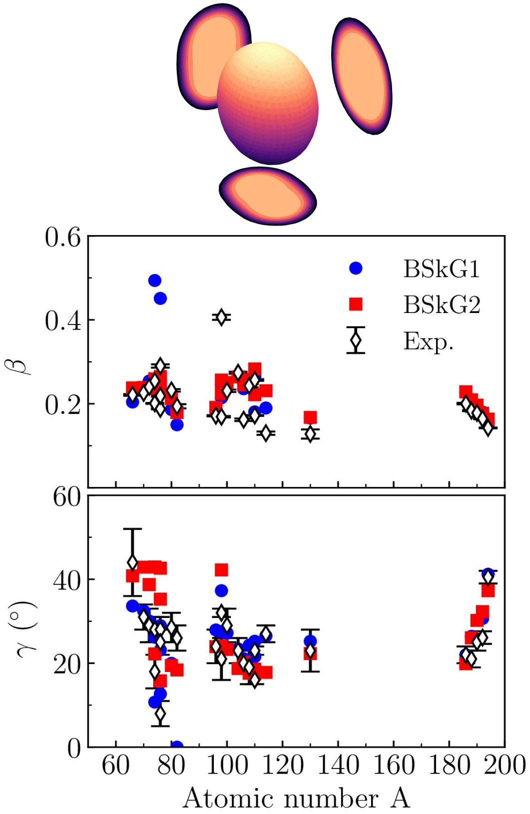
- small number of known nuclei affected
- Near N=184:
  - large effect up to 2.5 MeV
  - dripline modified
  - fission properties modified

Successes: deformations

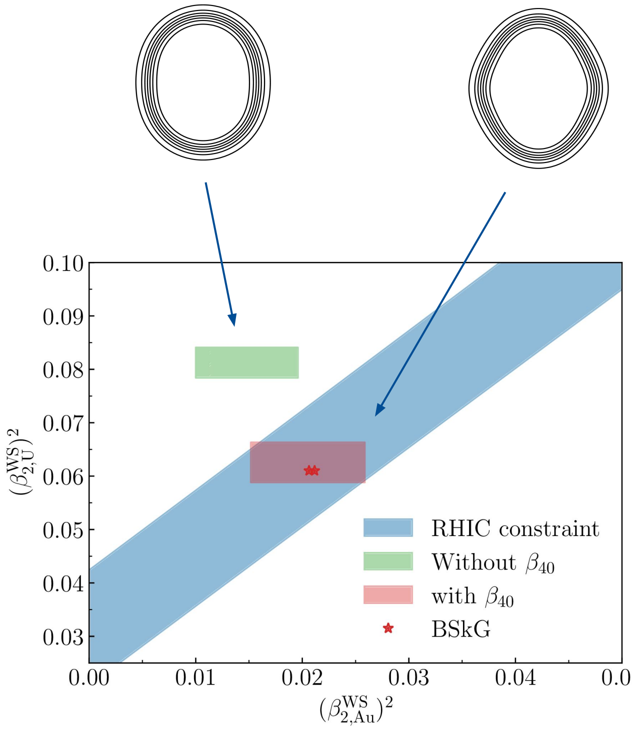
“Ordinary” quadrupole deformation



… and triaxial deformation …



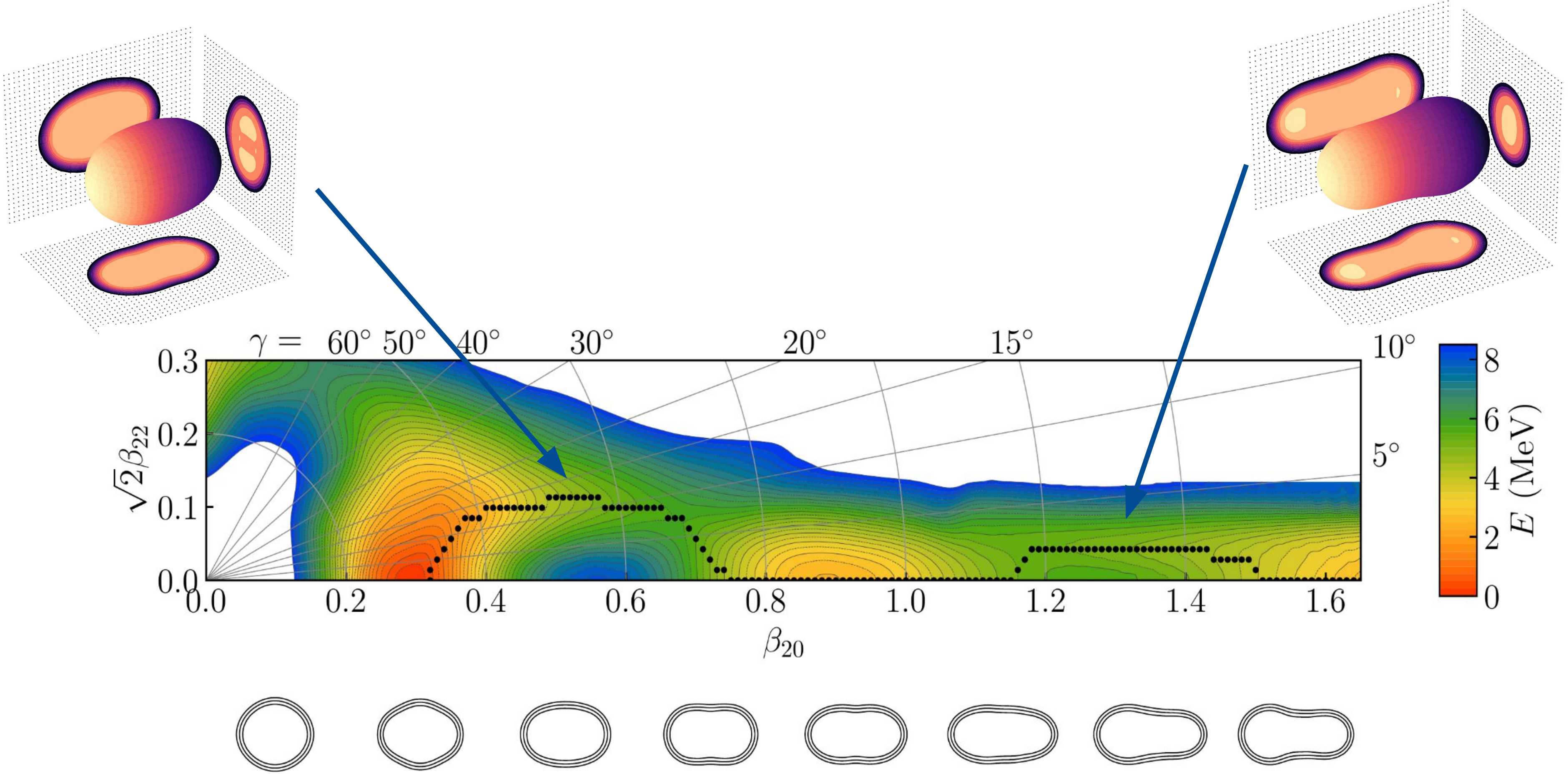
… and even hexadecapole!



W.R. et al., Phys. Rev. Lett. **130**, 212302

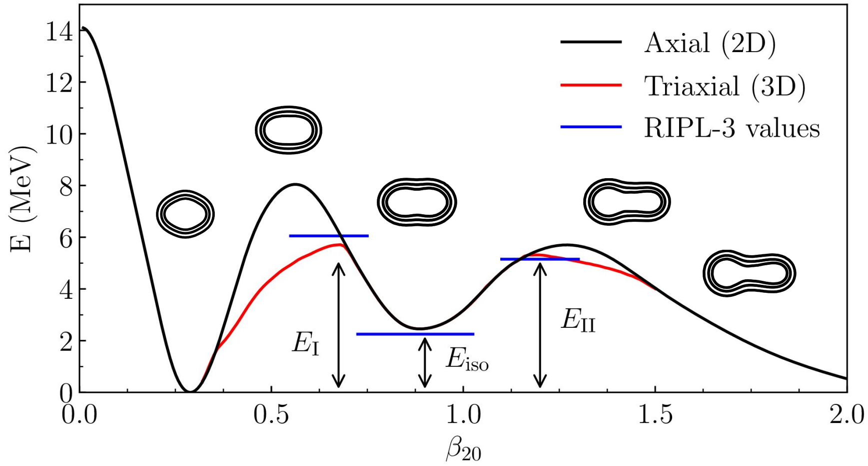
Successes: fission

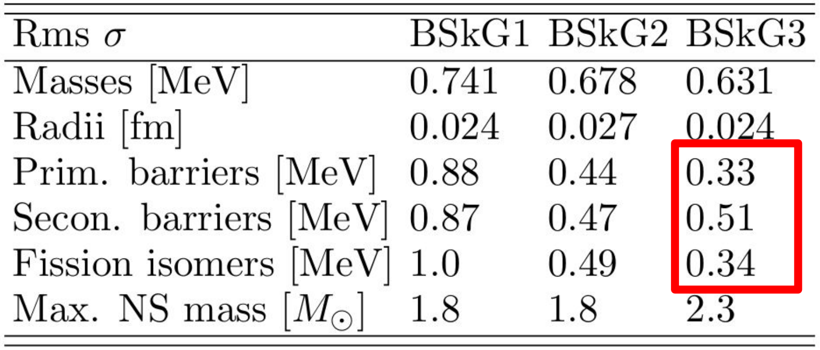
W. R. et al., EPJA **59**, 96 (2023).



Successes: fission

W. R. et al., EPJA **59**, 96 (2023).





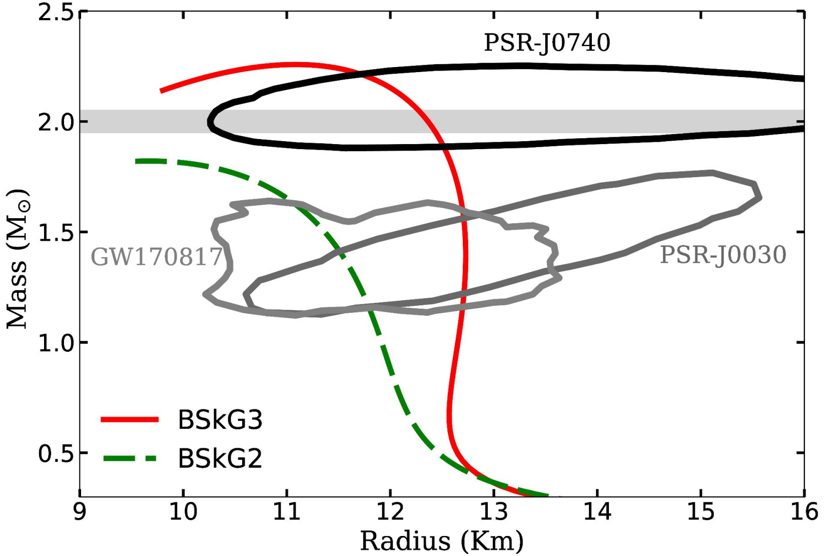
Rms σ	BSkG1	BSkG2	BSkG3
Masses [MeV]	0.741	0.678	0.631
Radii [fm]	0.024	0.027	0.024
Prim. barriers [MeV]	0.88	0.44	0.33
Secon. barriers [MeV]	0.87	0.47	0.51
Fission isomers [MeV]	1.0	0.49	0.34
Max. NS mass [$M_\odot$]	1.8	1.8	2.3

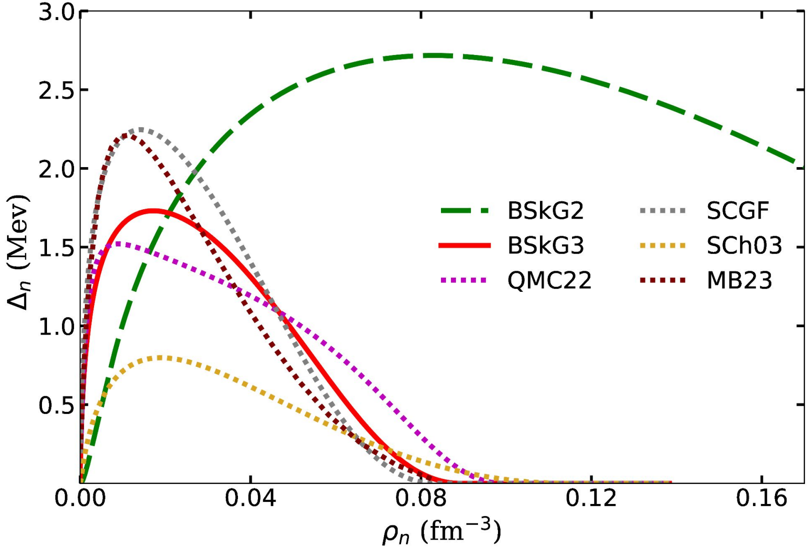
Fission properties of 45 actinide nuclei

- includes odd-A and odd-odds
- **all** inner barriers exploit triaxiality
- **all** outer barriers exploit
  - octupole deformation
  - triaxial deformation

Successes: neutron stars

G. Grams et al., EPJA 59, 270 (2023).





More realistic NS predictions:

- higher maximum mass
- realistic pairing properties in INM
  - constrained to advanced calculations
- …. but not at the cost of finite nuclei!
  - at the cost of extra density dependencies

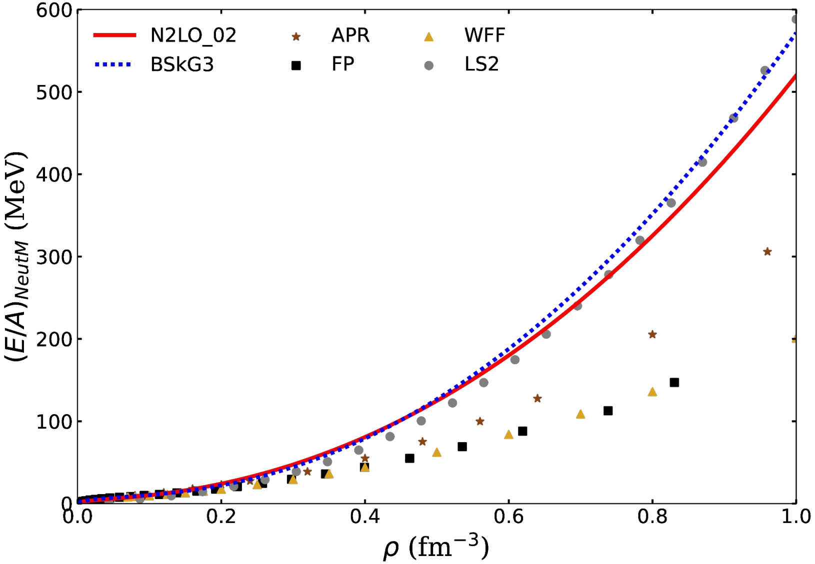
BSkG4: an N2LO parameterization?

Carlsson & Dobaczewski, PRC **78**, (2008).

N2LO

$$\hat{V}_{\mathrm{Sk}} = \boxed{\boxed{\hat{V}^{(0)}_{\mathrm{Sk}} + \hat{V}^{(2)}_{\mathrm{Sk}}} + \hat{V}^{(4)}_{\mathrm{Sk}}} + \hat{V}^{(6)}_{\mathrm{Sk}} .$$

NLO = traditional Skyrme



NxLO functional forms

- **systematic** expansion in gradients….
- …. but not improvable
- hope for improved spectroscopy
- significant numerical challenges

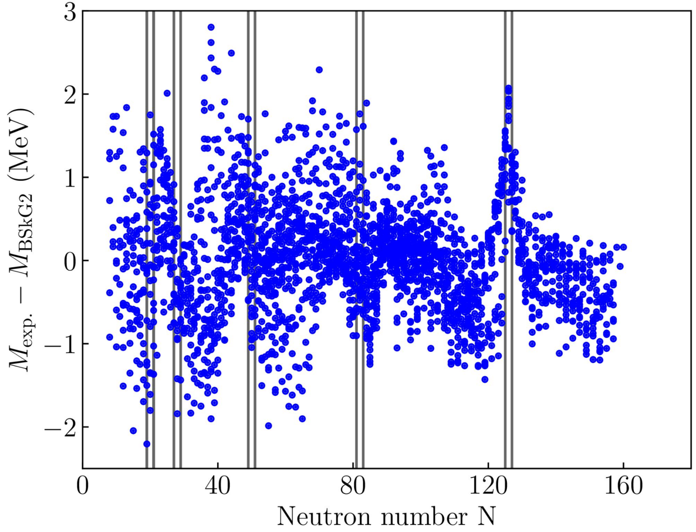
  => See MB's talk

My (modest) hopes

- realistic masses
- sufficiently heavy neutron stars
- sufficient DoF to get an effective mass more in line with ab initio indications
- … with LESS density dependencies

Failures

1. Residual structure in mass differences
2. Disastrous spectroscopy
3. Missing global observables
4. Phenomenological treatment of collective motion
5. Form of the functional



$$
\begin{aligned}
\mathcal{E}_{t,\mathrm{e}}(\mathbf{r}) = {} & C_t^{\rho\rho}(\rho_0)\,\rho_t^2(\mathbf{r}) + C_t^{\rho\tau}(\rho_0)\,\rho_t(\mathbf{r})\,\tau_t(\mathbf{r}) \\
& + C_t^{\rho\nabla J}\,\rho_t(\mathbf{r})\,\boldsymbol{\nabla}\cdot\mathbf{J}_t(\mathbf{r}) \\
& + C_t^{\rho\Delta\rho}\,\rho_t(\mathbf{r})\,\Delta\rho_t(\mathbf{r}) \\
& + C_t^{\nabla\rho\nabla\rho}(\rho_0)\,\boldsymbol{\nabla}\rho_t(\mathbf{r})\cdot\boldsymbol{\nabla}\rho_t(\mathbf{r}) \\
& + C_t^{\rho\nabla\rho\nabla\rho}(\rho_0)\,\rho_t(\mathbf{r})\boldsymbol{\nabla}\rho_0(\mathbf{r})\cdot\boldsymbol{\nabla}\rho_t(\mathbf{r}) \\
\mathcal{E}_{t,\mathrm{o}}(\mathbf{r}) = {} & C_t^{ss}(\rho_0)\,\mathbf{s}_t(\mathbf{r})\cdot\mathbf{s}_t(\mathbf{r}) + C_t^{jj}(\rho_0)\,\mathbf{j}_t(\mathbf{r})\cdot\mathbf{j}_t(\mathbf{r}) \\
& + C_t^{j\nabla s}\,\mathbf{j}_t(\mathbf{r})\cdot\nabla\times\mathbf{s}_t(\mathbf{r})\,.
\end{aligned}
$$

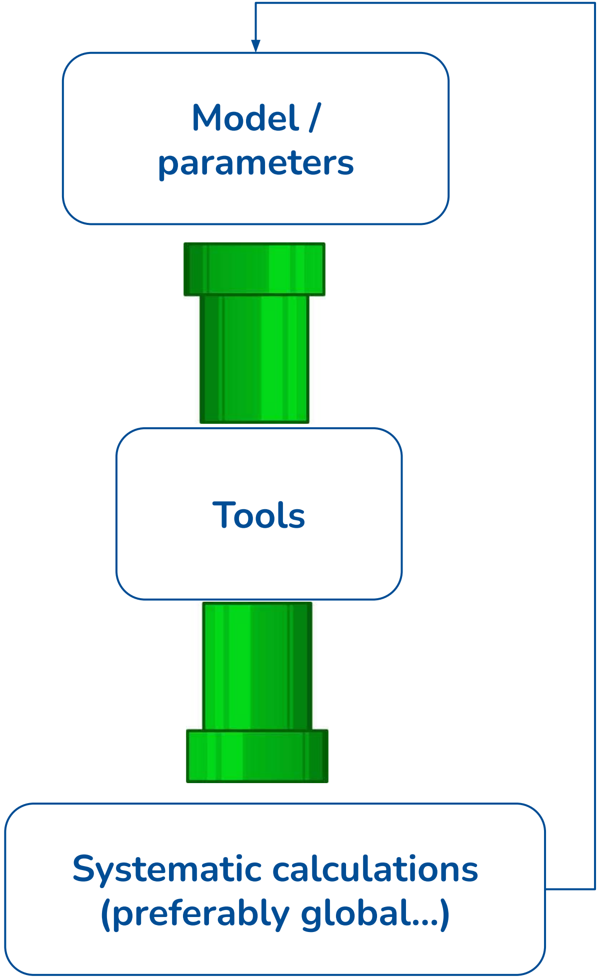
Table of contents

The pipeline is:

1. **incomplete**
    a. almost nothing is shared
    (exception: HFB solvers)
    b. not all nuclei are even-even
2. **fragile**
    a. interoperability ~ 0
    b. LOTS of oversight needed
    c. not repeatable
3. **slow**
    a. "good-enough" algorithms
    b. development takes forever

"How routine are global calculations?"

They are absolutely not!



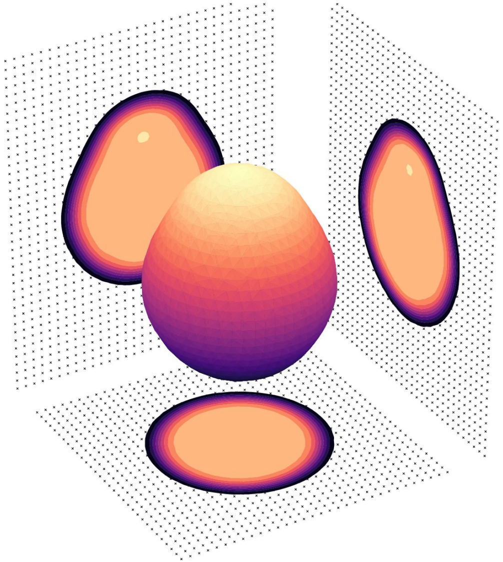
Model / parameters

Tools

Systematic calculations (preferably global…)

The tools behind: MOCCa

```
     ; ,
    ) ;( (
   ( (  ) ;
    ,_"""_.
 ,-|`-...-'|
((_|       |
 `-\       /
    `.___.'
```



W. R. PhD Thesis, ULB (2016).
W. R. et. al., PRC 92, 064318 (2015).
W. R. et. al., EPJA 55, 93 (2019).
W. R. and M. Herbst, in preparation.
W. R., in preparation.

HFB solver

- successor to EV8/EV4/CR8/...
- flexibility regarding imposed symmetries
- 3D coordinate space representation
- high and easily controllable accuracy

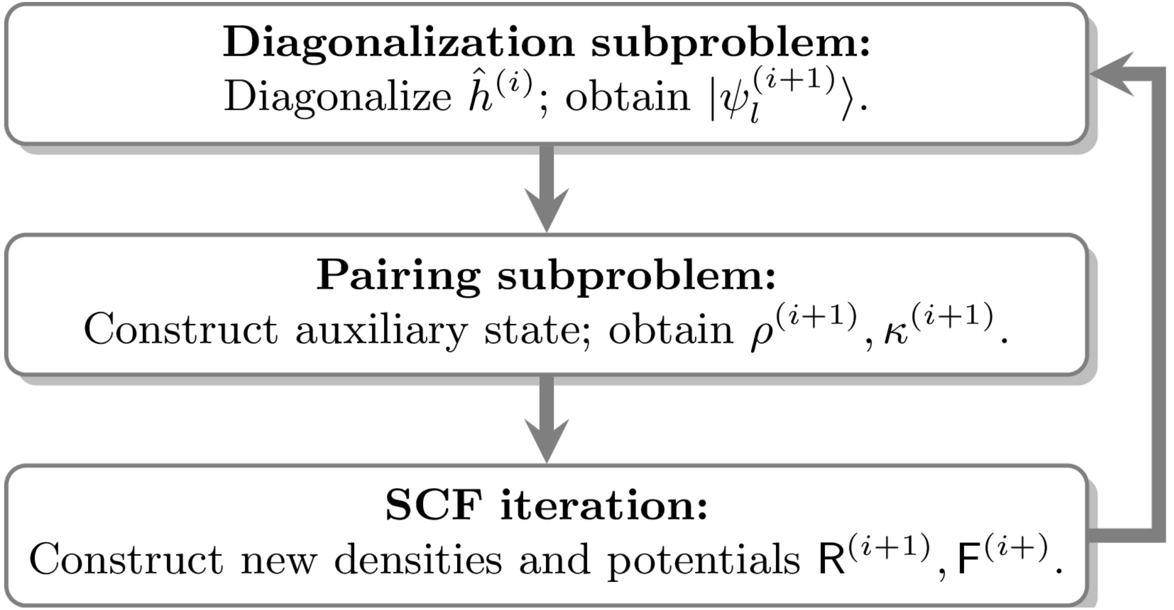
Algorithms

1. ... for speed
2. .... for EDF / spacing-agnostic convergence
3. ... to automatically estimate numerical parameters
4. .... for **automatic implementation** of EDFs

Coming to EPJA in **2024!**

MOCCa: algorithms

* W. Ryssens et al. EPJA **55** (2019).
# W. Ryssens, (forever) in preparation



Diagonalization subproblem:
Diagonalize $\hat{h}^{(i)}$; obtain $|\psi_l^{(i+1)}\rangle$.

Pairing subproblem:
Construct auxiliary state; obtain $\rho^{(i+1)}, \kappa^{(i+1)}$.

SCF iteration:
Construct new densities and potentials $\mathsf{R}^{(i+1)}, \mathsf{F}^{(i+)}$.

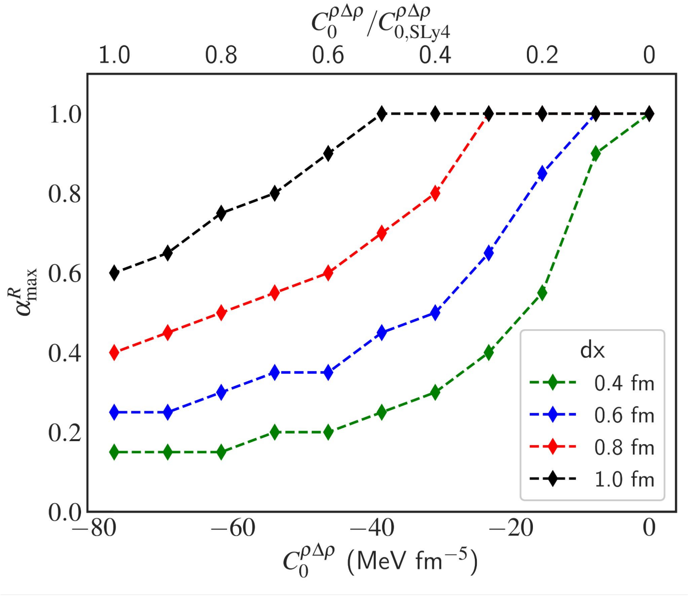
Heavy-ball iterations *

Gradient-pairing-solver **#**

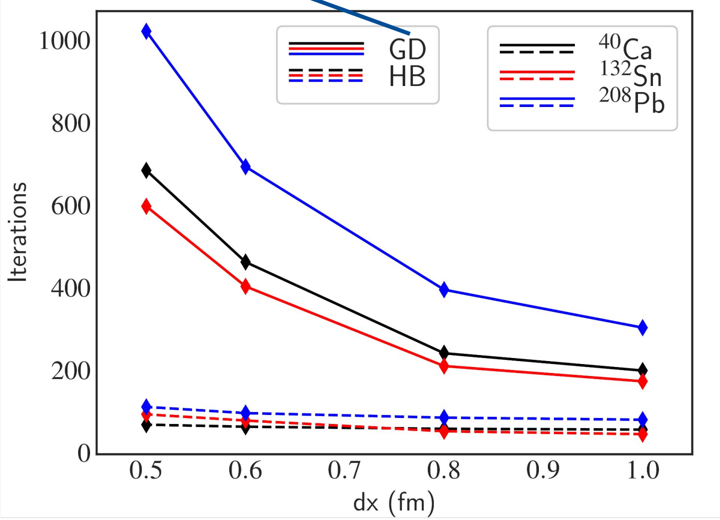
Potential preconditioning *
Two-step constraints **#**

MOCCa: algorithms

W. Ryssens et al. EPJA **55** (2019).



Gradient **D**escent
Heavy **B**all



Convergence depends on many things

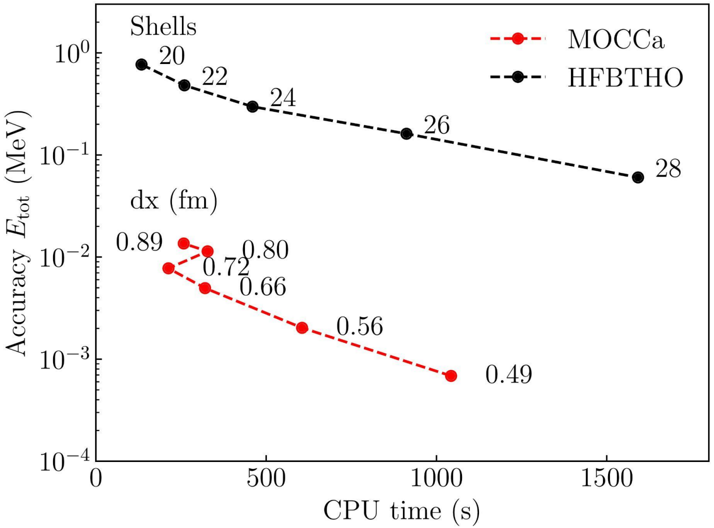
- details of representation
- balance of coupling constants
- nasty surprises in complicated EDF forms

Compared to EV8

- one order of magnitude speed-up
- with minimal coding
- ENTIRELY fire-and-forget
- [also algorithms for constraints)

MOCCa: algorithms

P. Marevic et al. CPC **276** (2022).



Comparison vs HFBTHO (v4.0)

- 2D HO versus 3D r-space
- HF minimum of ^{240}Pu for SLy4
- reference: 30 shells, dx = 0.43 fm

MOCCa: algorithms

L. Robledo & G. Bertsch, PRC **84**, 014312 (2012)
W. Ryssens, (forever) in preparation

$$
\begin{aligned}
\mathcal{H}(\mathcal{R})\mathcal{W} &= E^{\mathrm{qp}}\mathcal{W}\,, \\
\mathcal{R} &= \mathcal{W}^{\dagger}\mathcal{C}\mathcal{W}\,,
\end{aligned}
$$

$$
\delta R_{xy} \sim \sum_{\alpha\kappa ab} \frac{C_{\kappa} - C_{\alpha}}{E_{\kappa} - E_{\alpha}} W_{\alpha a} W^{\dagger}_{b\kappa} \mathcal{W}^{\dagger}_{x\alpha} \mathcal{W}_{\kappa y} \delta\mathcal{H}_{ab}\,.
$$

Direct diagonalisation

- easiest to code
- the choice of blocked qp is crucial
- doomed to fail in systematic calculations
- particularly with time-odd terms

$$
\begin{aligned}
\mathcal{W} &\to e^{-i\mathcal{Z}}\mathcal{W} \\
&\approx [1 - i\mathcal{Z}]\mathcal{W}\,.
\end{aligned}
$$

$$
\mathcal{Z} = \alpha \begin{pmatrix} 0 & H^{20} \\ H^{02} & 0 \end{pmatrix}
$$

$$
\delta\mathcal{R} = -i[\mathcal{Z}, \mathcal{R}]\,.
$$

Gradient solver

- based on Thouless theorem
- can also be accelerated by HB!
- almost unconditionally stable
- only lowest state accessible (for now...)

The tools behind: Hephaestos

Many different Skyrme EDF extensions:

- N2LO/N3LO
  B.G. Carlsson et al., PRC 78, 044326 (2008).
  B. G. Carlsson, PRL 105, 122501 (2010).
  D. Davesne et al. PRC 91, 064303. (2015).
- Tensor terms
  F. Tondeur , PLB 123, 139 (1983).
  T. Lesinski et al., PRC 76, 014312 (2007).
  G. Colò et al., PLB 646, 227 (2007).
- multi-reference
  J. Dobaczewski et al., NPA 422, 103 (1984).
  J. Dobaczewski, J. Phys. G 43, 04LT01 (2016).
  J.Sadoudi et al., Phys. Scripta, T154, 014013 (2013).
- density dependencies
  S.A. Fayans et al., NPA 676, 49–119 (2000).
  N. Chamel et al., PRC 80, 065804 (2009).
  A. Bulgac et al., PRC 97,044313 (2018).
  P.-G. Reinhard et al., PRC 95, 064328 (2017).
- ab initio

  [too many to list]

All of these forms are complicated...
... and none explored at scale!

This complexity stops both
1. initial exploration
2. sharing of EDF forms

$$
\begin{aligned}
\mathcal{E}^{(0)}_{\mathrm{Sk,e}}(\vec{r}) &= \sum_{t=0,1} \Big[A^{(0,1)}_{\mathrm{t,e}} \left(D^{1,1}_t\right)^2 + A^{(0,2)}_{\mathrm{t,e}} \left(D^{1,1}_0\right)^\alpha \left(D^{1,1}_t\right)^2 \Big] , \\
\mathcal{E}^{(2)}_{\mathrm{Sk,e}}(\vec{r}) &= \sum_{t=0,1} \Big[A^{(2,1)}_{\mathrm{t,e}} D^{1,1}_t \left(\Delta D^{1,1}_t\right) + A^{(2,2)}_{\mathrm{t,e}} D^{1,1}_t D^{(\nabla,\nabla)}_t + A^{(2,3)}_{\mathrm{t,e}} \sum_{\mu\nu} C^{1,\nabla\sigma}_{t,\mu\nu} C^{1,\nabla\sigma}_{t,\mu\nu} + A^{(2,4)}_{\mathrm{t,e}} D^{1,1}_t \left(\nabla \cdot C^{1,\nabla\times\sigma}_t\right) \Big] , \\
\mathcal{E}^{(4)}_{\mathrm{Sk,e}}(\vec{r}) &= \sum_{t=0,1} \Big[A^{(4,1)}_{\mathrm{t,e}} \left(\Delta D^{1,1}_t\right) \left(\Delta D^{1,1}_t\right) + A^{(4,2)}_{\mathrm{t,e}} D^{1,1}_t D^{\Delta,\Delta}_t + A^{(4,3)}_{\mathrm{t,e}} D^{(\nabla,\nabla)}_t D^{(\nabla,\nabla)}_t \\
&\quad + A^{(4,4)}_{\mathrm{t,e}} \sum_{\mu\nu} D^{\nabla,\nabla}_{t,\mu\nu} D^{\nabla,\nabla}_{t,\mu\nu} + A^{(4,5)}_{\mathrm{t,e}} \sum_{\mu\nu} D^{\nabla,\nabla}_{t,\mu\nu} \left(\nabla_\mu \nabla_\nu D^{1,1}_t\right) \\
&\quad + A^{(4,6)}_{\mathrm{t,e}} \sum_{\mu\nu} C^{1,\nabla\sigma}_{t,\mu\nu} \left(\Delta C^{1,\nabla\sigma}_{t,\mu\nu}\right) + A^{(4,7)}_{\mathrm{t,e}} \sum_{\mu\nu\kappa} \left(\nabla_\mu C^{1,\nabla\sigma}_{t,\mu\kappa}\right) \left(\nabla_\nu C^{1,\nabla\sigma}_{t,\nu\kappa}\right) + A^{(4,8)}_{\mathrm{t,e}} \sum_{\mu\nu} C^{1,\nabla\sigma}_{t,\mu\nu} C^{\Delta,\nabla\sigma}_{t,\mu\nu} \Big] , \\
\mathcal{E}^{(0)}_{\mathrm{Sk,o}}(\vec{r}) &= \sum_{t=0,1} \Big[A^{(0,1)}_{\mathrm{t,o}} \vec{D}^{1,\sigma}_t \cdot \vec{D}^{1,\sigma}_t + A^{(0,2)}_{\mathrm{t,o}} (D^{1,1}_0)^\alpha \vec{D}^{1,\sigma}_t \cdot \vec{D}^{1,\sigma}_t \Big] , \\
\mathcal{E}^{(2)}_{\mathrm{Sk,o}}(\vec{r}) &= \sum_{t=0,1} \Big[A^{(2,1)}_{\mathrm{t,o}} \vec{D}^{1,\sigma}_t \cdot \left(\Delta \vec{D}^{1,\sigma}_t\right) + A^{(2,2)}_{\mathrm{t,o}} \vec{D}^{1,\sigma}_t \cdot \vec{D}^{(\nabla,\nabla)\sigma}_t + A^{(2,3)}_{\mathrm{t,o}} \vec{C}^{1,\nabla}_t \cdot \vec{C}^{1,\nabla}_t + A^{(2,4)}_{\mathrm{t,o}} \vec{D}^{1,\sigma}_t \cdot \left(\boldsymbol{\nabla} \times \vec{C}^{1,\nabla}_t\right) \Big] , \\
\mathcal{E}^{(4)}_{\mathrm{Sk,o}}(\vec{r}) &= \sum_{t=0,1} \Big[A^{(4,1)}_{\mathrm{t,o}} \left(\Delta \vec{D}^{1,\sigma}_t\right) \cdot \left(\Delta \vec{D}^{1,\sigma}_t\right) + A^{(4,2)}_{\mathrm{t,o}} \vec{D}^{1,\sigma}_t \cdot \vec{D}^{\Delta,\Delta\sigma}_t + A^{(4,3)}_{\mathrm{t,o}} \vec{D}^{(\nabla,\nabla)\sigma}_t \cdot \vec{D}^{(\nabla,\nabla)\sigma}_t \\
&\quad + A^{(4,4)}_{\mathrm{t,o}} \sum_{\mu\nu\kappa} D^{\nabla,\nabla\sigma}_{\mu\nu\kappa} D^{\nabla,\nabla\sigma}_{\mu\nu\kappa} + A^{(4,5)}_{\mathrm{t,o}} \sum_{\mu\nu\kappa} D^{\nabla,\nabla\sigma}_{\mu\nu\kappa} \left(\nabla_\mu \nabla_\nu D^{1,\sigma}_\kappa\right) , \\
&\quad + A^{(4,6)}_{\mathrm{t,o}} \vec{C}^{1,\nabla}_t \cdot \left(\Delta \vec{C}^{1,\nabla}_t\right) + A^{(4,7)}_{\mathrm{t,o}} \left(\boldsymbol{\nabla} \cdot \vec{C}^{1,\nabla}_t\right) \left(\boldsymbol{\nabla} \cdot \vec{C}^{1,\nabla}_t\right) + A^{(4,8)}_{\mathrm{t,o}} \vec{C}^{1,\nabla}_t \cdot \vec{C}^{\Delta,\nabla}_t \Big] ,
\end{aligned}
$$

W. Ryssens & M. Bender, PRC **104** 044308 (2021).

The tools behind: Hephaestos

W. Ryssens & M. Bender, **104,** 044308 (2021).

$$D^{L,R}(\mathbf{r}) = \mathrm{Re}\left\{\hat{L}(\mathbf{r}')\hat{R}(\mathbf{r})\rho(\mathbf{r},\mathbf{r}')\right\}\Big|_{\mathbf{r}=\mathbf{r}'} ,$$

$$C^{L,R}(\mathbf{r}) = \mathrm{Im}\left\{\hat{L}(\mathbf{r}')\hat{R}(\mathbf{r})\rho(\mathbf{r},\mathbf{r}')\right\}\Big|_{\mathbf{r}=\mathbf{r}'} ,$$

$$\rho(\mathbf{r}) \rightarrow \quad D^{1,1}(\mathbf{r}) = \mathtt{D\_I\_I} \quad ,$$

$$\mathbf{j}(\mathbf{r}) \rightarrow \quad \mathbf{C}^{1,\nabla}(\mathbf{r}) = \mathtt{C\_I\_N} \quad ,$$

$$\tau(\mathbf{r}) \rightarrow D^{(\nabla,\nabla)}(\mathbf{r}) = \mathtt{D\_Nm\_Nm} ,$$

$$\mathsf{R}_q = \big(D_q^{1,1},\, D_{q,\mu}^{1,\sigma},\, D_{q,\mu\nu}^{\nabla,\nabla},\, C_{q,\mu\nu}^{1,\nabla\sigma},\, C_{q,\mu}^{1,\nabla},\, D_{q,\mu\nu\kappa}^{\nabla,\nabla\sigma},$$
$$D_q^{\Delta,\Delta},\, C_{q,\mu\nu}^{\Delta,\nabla\sigma},\, D_{q,\mu}^{\Delta,\Delta\sigma},\, C_{q,\mu}^{\Delta,\nabla}\big).$$

$$\mathsf{F}_{q,a}(\mathbf{r}) \equiv \frac{\delta E_{\mathrm{tot}}(\mathsf{R})}{\delta \mathsf{R}_{q,a}(\mathbf{r})}.$$

Automating EDFs

A HFB solver only depends on the EDF-type via:

1. construction of densities
2. construction of mean-fields (ph and pp)

Formulas for all relevant quantities obtained through many applications of simple rules.

Complex for humans, but easy for computers!

$$\rho\tau \rightarrow \mathtt{D\_I\_I\_D\_Nm\_Nm}$$

$$F_\rho(\mathbf{r}) \rightarrow \frac{\partial E}{\partial \rho} \sim \mathtt{D\_Nm\_Nm} ,$$

$$F_\tau(\mathbf{r}) \rightarrow \frac{\partial E}{\partial \rho} \sim \mathtt{D\_I\_I} ,$$

The tools behind: Hephaestos

Hephaestos automates EDF implementation!

Current state:

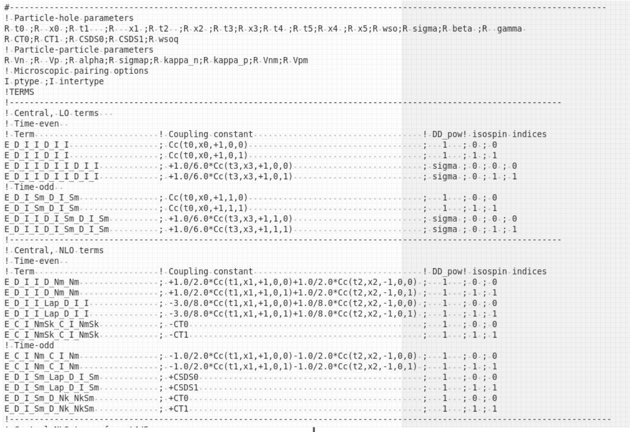
- arbitrary densities up to N3LO
- no limit on the densities in a term
- (almost) arbitrary density dependencies
- writes quite efficient code
- functional file writing remains a bit complex

Long-term goals:

- extend applications (INM, QRPA, MR, ...)
- starting from a LaTeX expression
- make things solver-agnostic!

This is the tool behind:

- BSkG3: time-to-start-fitting ~ **1 week**
- BSkG4: time-to-start-fitting ~ **1 day**



```
#-------------------------------------------------------------------------------------------------------------
! Particle-hole parameters
R t0 ;R  x0 ;R t1   ;R   x1 ;R t2  ;R x2 ;R t3;R x3;R t4 ;R t5;R x4 ;R x5;R wso;R sigma;R beta ;R  gamma
R CT0;R CT1 ;R CSDS0;R CSDS1;R wsoq
! Particle-particle parameters
R Vn ;R  Vp ;R alpha;R sigmap;R kappa_n;R kappa_p;R Vnm;R Vpm
! Microscopic pairing options
I ptype ;I intertype
!TERMS
!---------------------------------------------------------------------------------------------------
! Central, LO terms
! Time-even
! Term                         ! Coupling constant                                    ! DD_pow! isospin indices
E_D_I_I_D_I_I                  ; Cc(t0,x0,+1,0,0)                                     ;   1   ; 0 ; 0
E_D_I_I_D_I_I                  ; Cc(t0,x0,+1,0,1)                                     ;   1   ; 1 ; 1
E_D_I_I_D_I_I_D_I_I            ; +1.0/6.0*Cc(t3,x3,+1,0,0)                            ; sigma ; 0 ; 0 ; 0
E_D_I_I_D_I_I_D_I_I            ; +1.0/6.0*Cc(t3,x3,+1,0,1)                            ; sigma ; 0 ; 1 ; 1
! Time-odd
E_D_I_Sm_D_I_Sm                ; Cc(t0,x0,+1,1,0)                                     ;   1   ; 0 ; 0
E_D_I_Sm_D_I_Sm                ; Cc(t0,x0,+1,1,1)                                     ;   1   ; 1 ; 1
E_D_I_I_D_I_Sm_D_I_Sm          ; +1.0/6.0*Cc(t3,x3,+1,1,0)                            ; sigma ; 0 ; 0 ; 0
E_D_I_I_D_I_Sm_D_I_Sm          ; +1.0/6.0*Cc(t3,x3,+1,1,1)                            ; sigma ; 0 ; 1 ; 1
!---------------------------------------------------------------------------------------------------
! Central, NLO terms
! Time-even
! Term                         ! Coupling constant                                    ! DD_pow! isospin indices
E_D_I_I_D_Nm_Nm                ; +1.0/2.0*Cc(t1,x1,+1,0,0)+1.0/2.0*Cc(t2,x2,-1,0,0)  ;   1   ; 0 ; 0
E_D_I_I_D_Nm_Nm                ; +1.0/2.0*Cc(t1,x1,+1,0,1)+1.0/2.0*Cc(t2,x2,-1,0,1)  ;   1   ; 1 ; 1
E_D_I_I_Lap_D_I_I              ; -3.0/8.0*Cc(t1,x1,+1,0,0)+1.0/8.0*Cc(t2,x2,-1,0,0)  ;   1   ; 0 ; 0
E_D_I_I_Lap_D_I_I              ; -3.0/8.0*Cc(t1,x1,+1,0,1)+1.0/8.0*Cc(t2,x2,-1,0,1)  ;   1   ; 1 ; 1
E_C_I_NmSk_C_I_NmSk            ; -CT0                                                 ;   1   ; 0 ; 0
E_C_I_NmSk_C_I_NmSk            ; -CT1                                                 ;   1   ; 1 ; 1
! Time-odd
E_C_I_Nm_C_I_Nm                ; -1.0/2.0*Cc(t1,x1,+1,0,0)-1.0/2.0*Cc(t2,x2,-1,0,0)  ;   1   ; 0 ; 0
E_C_I_Nm_C_I_Nm                ; -1.0/2.0*Cc(t1,x1,+1,0,1)-1.0/2.0*Cc(t2,x2,-1,0,1)  ;   1   ; 1 ; 1
E_D_I_Sm_Lap_D_I_Sm            ; +CSDS0                                               ;   1   ; 0 ; 0
E_D_I_Sm_Lap_D_I_Sm            ; +CSDS1                                               ;   1   ; 1 ; 1
E_D_I_Sm_D_Nk_NkSm             ; +CT0                                                 ;   1   ; 0 ; 0
E_D_I_Sm_D_Nk_NkSm             ; +CT1                                                 ;   1   ; 1 ; 1
!-------------------------------------------------------------------------------------------------------------
```

```
 _   _            _                     _
| | | | ___ _ __ | |__   __ _  ___  ___| |_ ___  ___
| |_| |/ _ \ '_ \| '_ \ / _` |/ _ \/ __| __/ _ \/ __|
|  _  |  __/ |_) | | | | (_| |  __/\__ \ || (_) \__ \
|_| |_|\___| .__/|_| |_|\__,_|\___||___/\__\___/|___/
           |_|
```

```
     ; ,
    ) ;( (
   ( (  ) ;
    ,_"""_.
 ,-|`-...-'|
((_|       |
 `-\       /
    `     '
     .___.
```

Table of contents

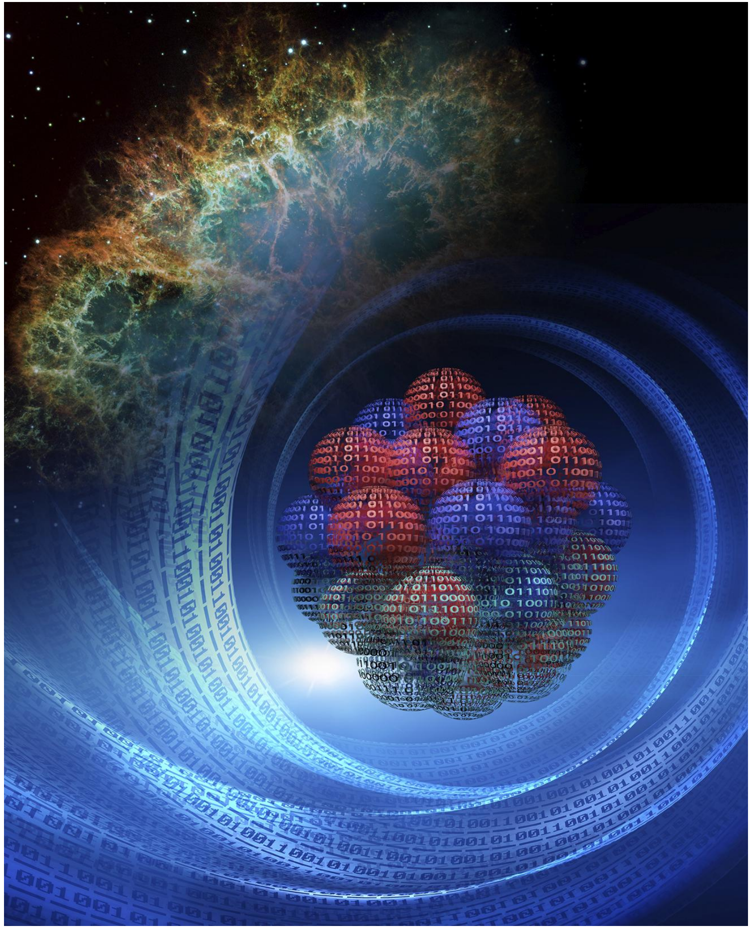
Conclusion

BSkG-models

- symmetry breaking to the max
- excellent bulk properties
- exploitation ongoing
- N2LO version in the works

Numerics

- our tools deserve attention…
- … automation will be key.
- MOCCa is robust and fast….
- … and will go open-source in 2024!



What I will be working on:

Improving the BSkG-models:

1. towards H's with (ab initio?) inspiration
   - which form to start with?
2. larger reach in terms of observables
   - FAM-QRPA, large-scale fission, ...
3. treatment of collective motion
   - towards VAP but first through FAM-QRPA

One step at a time!

Crucial in this will be "fixing" the pipeline:

1. completeness
2. robustness
3. speed

Thank you for...

...... all the wonderful work!

ULB	S. Goriely G. Grams N. Chamel N. Shchechilin	iP2i LES 2 INFINIS LYON	M. Bender J. Bonnard
L2IT	G. Scamps		
JYVÄSKYLÄN YLIOPISTO UNIVERSITY OF JYVÄSKYLÄ	M. Hukkanen M. Stryjczyk A. Kankainen	LP2i Bordeaux	P. Ascher S. Grévy
KU LEUVEN	E. Verstraelen T. Cocolios P. Van Duppen		
UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386	G. Giacalone	Brookhaven National Laboratory	B. Schenke C. Shen
cea	S. Hilaire		

...... the computing time!

CÉCI

Cenaero

LUMI

...... the funding!

fnrs
LA LIBERTÉ DE CHERCHER

fwo

EOS
THE EXCELLENCE
OF SCIENCE

Thank **you** for…

… your attention!

… a decade of support!