Cosmo$\mathcal{L}$attice school:

Lesson 2b: Primer on Lattice simulations:
phi^4 in an expanding background

Daniel G. Figueroa
IFIC UV/CSIC, Spain

Adrien Florio
Stony Brook U., USA

Francisco Torrenti
U. Basel, Switzerland

Cosmo$\mathcal{L}$attice school, IFIC Valencia - 5th-8th September 2022

Table of contents

Program variables

- In the lattice we operate in a set of dimensionless spacetime and field variables called **program variables**:

Choose:

$$\{\alpha, f_*, \omega_*\} \longrightarrow \boxed{d\tilde{\eta} \equiv a^{-\alpha}\omega_* dt \quad d\tilde{x}^i \equiv \omega_* dx^i} \qquad \boxed{\tilde{\phi} = \frac{\phi}{f_*}}$$

Number

Mass scales

Space and time

Scalar fields

$\dot{f} \equiv df/dt$ denotes derivatives with respect to cosmic time.

$f' \equiv df/d\tilde{\eta}$ denotes derivatives with respect to program time.

- We also build other (dimensionless) quantities from program variables for convenience. These are tagged with the diacritic "~".

Example: Program potential

$$\boxed{\tilde{V}(\tilde{\phi}) = \frac{1}{f_*^2\omega_*^2} V(f_*\tilde{\phi})}$$

Program fields

Physical fields

Program variables

- Program variables are chosen **in a case by case basis**.

- Optimal choice for **monomial inflationary potentials** is:

$$\boxed{V(\phi) = \frac{1}{p}\lambda\mu^{4-p}|\phi|^p} \longrightarrow \begin{cases} \alpha = 3\left(\dfrac{p-2}{p+2}\right) \\ f_* = \phi_* \\ \omega_* = \sqrt{\lambda}\mu^{\frac{4-p}{2}}\phi_*^{\frac{p-2}{2}} \end{cases}$$

← (Cosmic time for p=2, conformal time for p=4)

← Amplitude at the end of inflation

← Oscillation frequency at the end of inflation

in these variables, the **oscillation frequency is approximately constant**, and **time and space scales are of order one.**

Example model: preheating in λφ⁴

- **EXAMPLE MODEL:** Inflaton with quartic potential, coupled to a "daughter" field through a quadratic-quadratic interaction term

$$S = \int d^4x a(t)^3 \mathcal{L}_m$$

$$\boxed{-\mathcal{L}_m = \frac{1}{2}\partial^\mu\phi\partial_\mu\phi + \frac{1}{2}\partial^\mu\chi\partial_\mu\chi + V(\phi,\chi)}$$

$$\boxed{V(\phi,\chi) \equiv \sum_{m=0}^{N_p-1} V^{(m)}(\phi,\chi) = \frac{\lambda}{4}\phi^4 + \frac{1}{2}g^2\phi^2\chi^2}$$

- Number scalar singlets: $N_s = 2$
- Number potential terms: $N_t = 2$

- We fix the **program variables** to:

$$\boxed{\begin{aligned} f_* &= \phi_* \\ \omega_* &= \sqrt{\lambda}\phi_* \\ \alpha &= 1 \end{aligned}} \longrightarrow \boxed{\tilde{\phi} = \frac{\phi}{\phi_*} \qquad d\tilde{\eta} \equiv \sqrt{\lambda}\phi_* a^{-1} dt \qquad d\tilde{x}^i \equiv \sqrt{\lambda}\phi_* dx^i}$$

[Note: unlike the "natural variables" introduced before, the program inflaton is not rescaled by the scale factor]

Example model: preheating in λφ⁴

- **Program potential:**

$$\boxed{\tilde{V}(\tilde{\phi},\tilde{\chi}) \equiv \frac{1}{f_*^2\omega_*^2}V(f_*\tilde{\phi},f_*\tilde{\chi}) = \overset{\tilde{V}^{(0)}}{\overset{\downarrow}{\frac{1}{4}|\tilde{\phi}|^4}} + \overset{\tilde{V}^{(1)}}{\overset{\downarrow}{\frac{1}{2}q\tilde{\phi}^2\tilde{\chi}^2}}}$$

$$\boxed{q \equiv \frac{g^2}{\lambda}}$$

Resonance parameter

- Analogously, we define the **program energy density** and **program pressure density:**

$$\boxed{\tilde{\rho} \equiv \frac{\rho}{f_*^2\omega_*^2} = \widetilde{K} + \widetilde{G} + \widetilde{V}} \qquad \boxed{\tilde{p} \equiv \frac{p}{f_*^2\omega_*^2} = \widetilde{K} - \frac{1}{3}\widetilde{G} - \widetilde{V}}$$

Kinetic contribution:

$$\widetilde{K} \equiv \sum_{n=0}^{N_s-1}\widetilde{K}^{(n)} \qquad \widetilde{K}^{(n)} \equiv \frac{1}{2a^{2\alpha}}(\tilde{\phi}_n^{\,\prime})^2 \qquad \xrightarrow{\hspace{2em}} \qquad \widetilde{E}_K \equiv \langle\tilde{K}\rangle$$

Volume averages

Gradient contribution:

$$\widetilde{G} \equiv \sum_{n=0}^{N_s-1}\widetilde{G}^{(n)} \qquad \widetilde{G}^{(n)} \equiv \frac{1}{2a^2}\sum_i(\widetilde{\nabla}_i\tilde{\phi}_n)^2 \qquad \xrightarrow{\hspace{2em}} \qquad \widetilde{E}_G \equiv \langle\tilde{G}\rangle$$

Potential contribution:

$$\widetilde{V} \equiv \sum_{n=0}^{N_t-1}\widetilde{V}^{(t)} \qquad \widetilde{V}^{(t)} \equiv \frac{1}{f_*^2\omega_*^2}V^{(t)}(\tilde{\phi}_n) \qquad \xrightarrow{\hspace{2em}} \qquad \widetilde{E}_V \equiv \langle\tilde{V}\rangle$$

Example model: preheating in λφ⁴

- The code solves the following equations:

$$\begin{aligned}\tilde{\phi}'' - a^{-2(1-\alpha)}\tilde{\nabla}^2\tilde{\phi} + (3-\alpha)\frac{a'}{a}\tilde{\phi}'_a &= -a^{2\alpha}\widetilde{V}_{,\tilde{\phi}}\\ &= -a^{2\alpha}\left(\tilde{\phi}^3 + \frac{g^2}{\lambda}\tilde{\phi}\tilde{\chi}^2\right)\end{aligned}$$

FIELD 0:

2*N³ equations

$$\begin{aligned}\chi'' - a^{-2(1-\alpha)}\tilde{\nabla}^2\tilde{\chi} + (3-\alpha)\frac{a'}{a}\tilde{\chi}'_a &= -a^{2\alpha}\widetilde{V}_{,\tilde{\chi}}\\ &= -a^{2\alpha}\frac{g^2}{\lambda}\tilde{\phi}^2\tilde{\chi}\end{aligned}$$

FIELD 1:

2*N³ equations

SCALE FACTOR:

1 equation

$$\frac{a''}{a} = \frac{a^{2\alpha}}{3}\left(\frac{f_*}{m_p}\right)^2\left[(\alpha-2)\widetilde{E}_K + \alpha\widetilde{E}_G + (\alpha+1)\widetilde{E}_V\right]$$

- The accuracy of our solution is monitored through the 1st Friedmann eq.:

$$a'^2 = \frac{a^{2\alpha+2}}{3}\left(\frac{f_*}{m_p}\right)^2\left[\widetilde{E}_K + \widetilde{E}_G + \widetilde{E}_V\right] \longrightarrow \Delta_E \equiv \frac{|\mathrm{LHS}-\mathrm{RHS}|}{|\mathrm{LHS}+\mathrm{RHS}|} \lesssim 1$$

Example model: preheating in λφ⁴

- The code solves the following equations:

$$\tilde{\phi}'' - a^{-2(1-\alpha)}\tilde{\nabla}^2\tilde{\phi} + (3-\alpha)\frac{a'}{a}\tilde{\phi}'_a = -a^{2\alpha}\widetilde{V}_{,\tilde{\phi}}$$

FIELD 0:

TOMORROW (Tuesday 6th Sept):

- **Lesson 3 [Adrien]:** Algorithms to solve 2nd order differential equations
- **Lesson 4 [Dani]:** Application of these algorithms to solve the EOM of interacting scalar fields

- The accuracy of our solution is monitored through the 1st Friedmann eq.:

$$a'^2 = \frac{a^{2\alpha+2}}{3}\left(\frac{f_*}{m_p}\right)^2\left[\widetilde{E}_K + \widetilde{E}_G + \widetilde{E}_V\right] \longrightarrow \Delta_E \equiv \frac{|\mathrm{LHS}-\mathrm{RHS}|}{|\mathrm{LHS}+\mathrm{RHS}|} \lesssim 1$$

Initial conditions in CosmoLattice

- Initial fluctuations in the continuum (in program variables)

$$\left\langle \delta\tilde\phi^2 \right\rangle = \frac{1}{2\pi^2}\int d\log\tilde k\ \tilde k^3\,\widetilde{\mathcal{P}}_{\delta\phi}(\tilde k)$$

$$\boxed{\widetilde{\mathcal{P}}_{\delta\tilde\phi}(\tilde k) \equiv \frac{1}{2a^2\widetilde\omega_{\tilde k,\tilde\phi}} \qquad \widetilde\omega_{k,\phi} \equiv \sqrt{\tilde k^2 + a^2\tilde m^2_{\tilde\phi}} \qquad \tilde m^2_{\phi} \equiv \frac{\partial^2\widetilde V}{\partial\tilde\phi^2}\bigg|_{\tilde\phi=\bar\phi_*}}$$

- Initial fluctuations in the lattice (in program variables):

$$\left\langle \delta\widetilde{\phi}^{\,2} \right\rangle_V = \frac{1}{2\pi^2}\sum_{|\tilde{\mathbf{n}}|}\Delta\log\tilde k(\tilde{\mathbf{n}})\ \tilde k^3(\tilde{\mathbf{n}})\left(\frac{\delta\tilde x}{N}\right)^3\left\langle \left|\delta\tilde\phi(\tilde{\mathbf{n}})\right|^2\right\rangle_{R(\tilde{\mathbf{n}})} \longrightarrow \left\langle \left|\delta\phi(\tilde{\mathbf{n}})\right|^2\right\rangle_{R(\tilde{\mathbf{n}})} \equiv \left(\frac{\omega_*}{f_*}\right)^2\left(\frac{N}{\delta\tilde x}\right)^3\widetilde{\mathcal{P}}_{\delta\tilde\phi}(\tilde k(\tilde{\mathbf{n}}))$$

(*)

- Fluctuations in each node are imposed as a sum of left- and right-moving waves:

$$\boxed{\begin{aligned}\delta\tilde\phi(\tilde{\mathbf{n}}) &= \frac{1}{\sqrt2}\left(|\,\delta\tilde\phi^{(l)}(\tilde{\mathbf{n}})\,|\,e^{i\theta^{(l)}(\tilde{\mathbf{n}})} + |\,\delta\tilde\phi^{(r)}(\tilde{\mathbf{n}})\,|\,e^{i\theta^{(r)}(\tilde{\mathbf{n}})}\right)\\ \delta\tilde\phi'(\tilde{\mathbf{n}}) &= \frac{1}{a^{1-\alpha}}\left[\frac{i\tilde\omega_{k,\phi}}{\sqrt2}\left(|\,\delta\tilde\phi^{(l)}(\tilde{\mathbf{n}})\,|\,e^{i\theta^{(l)}(\tilde{\mathbf{n}})} - |\,\delta\tilde\phi^{(r)}(\tilde{\mathbf{n}})\,|\,e^{i\theta^{(r)}(\tilde{\mathbf{n}})}\right)\right] - \tilde{\mathcal{H}}\,\delta\tilde\phi(\tilde{\mathbf{n}})\end{aligned}}$$

→ GAUSSIAN FLUCTUATIONS

$|\,\delta\tilde\phi^{(l,r)}(\tilde{\mathbf{n}})\,|$: Rayleigh distribution with expected **(*)**

$\theta^{(l,r)}(\tilde{\mathbf{n}})$: Random phase in range [0,2π]

$$\tilde{\mathcal{H}} \equiv a^\alpha H/\omega_*$$

CosmoLattice installation

- **Prerequisites (OS X or linux):**
  - CMake v3.0 (or above)
  - g++ v5.0 (or above) or clang++ v3.4 (or above)
  - fftw3 (note: it can be installed from our script)
  - *For parallel use:* MPI.
  - *Optional:* HDF5 and PFFT

More info on Appendix A of user manual!

- **Download the code:** Type in the terminal:

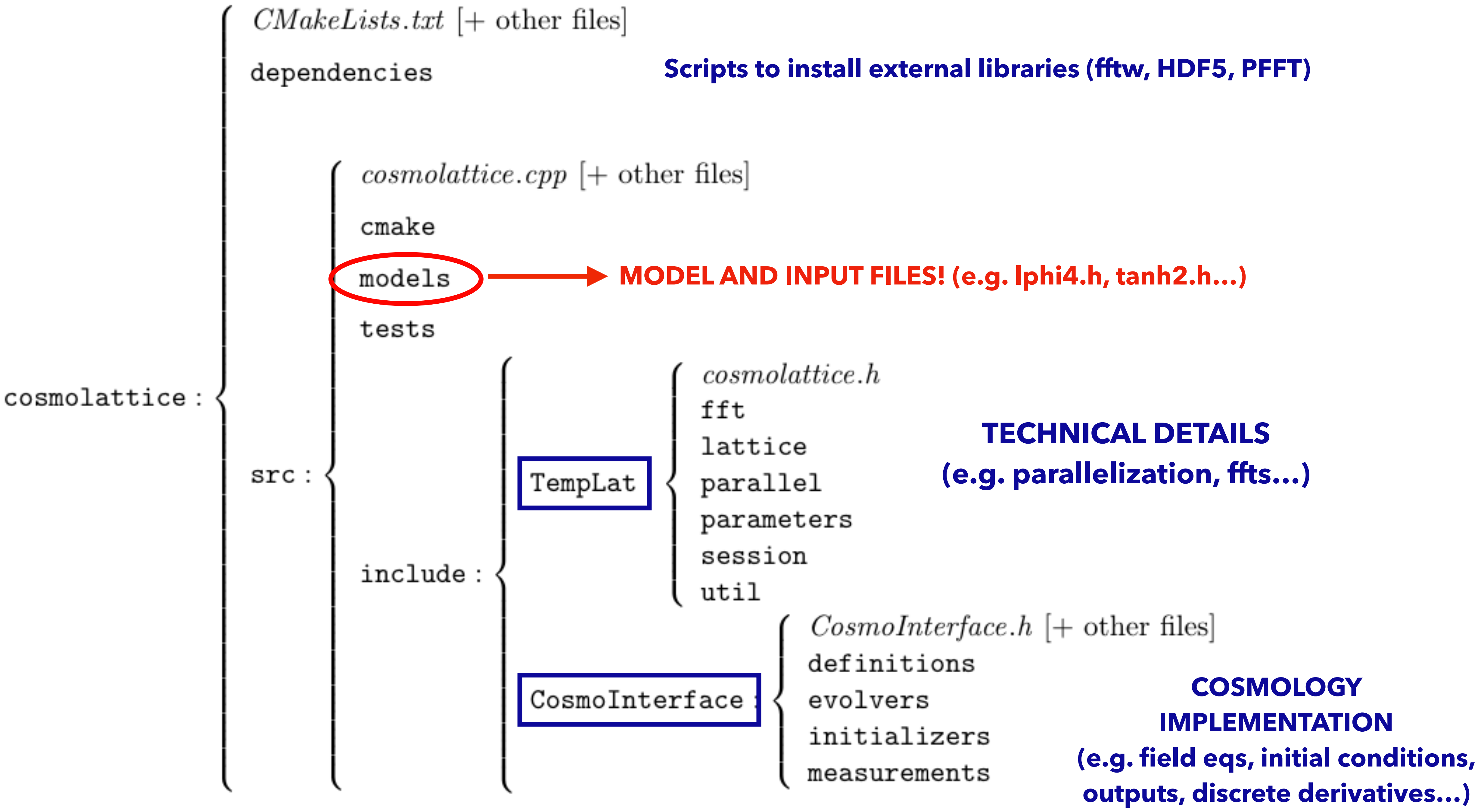
```
git clone https://github.com/cosmolattice/cosmolattice
```

(or download it directly from the link)

Implementation of a model only requires handling two files:
- **The model file**
- **The parameter "input" file**

CL folder tree structure

- Basic folder tree structure:



- cosmolattice :
  - *CMakeLists.txt* [+ other files]
  - dependencies
  - src :
    - *cosmolattice.cpp* [+ other files]
    - cmake
    - models
    - tests
    - include :
      - TempLat
        - *cosmolattice.h*
        - fft
        - lattice
        - parallel
        - parameters
        - session
        - util
      - CosmoInterface :
        - *CosmoInterface.h* [+ other files]
        - definitions
        - evolvers
        - initializers
        - measurements

Scripts to install external libraries (fftw, HDF5, PFFT)

MODEL AND INPUT FILES! (e.g. lphi4.h, tanh2.h…)

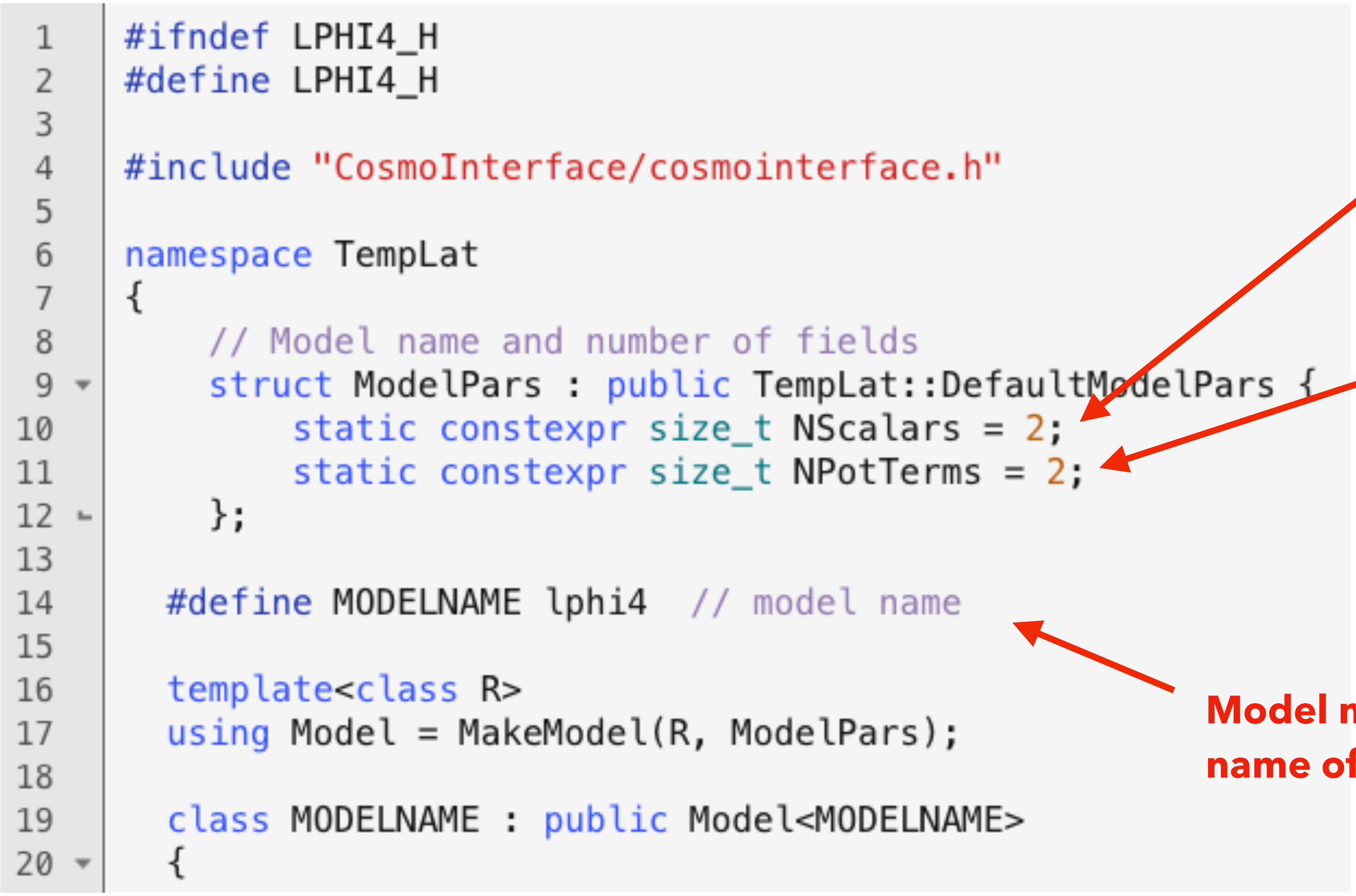
TECHNICAL DETAILS
(e.g. parallelization, ffts…)

COSMOLOGY
IMPLEMENTATION
(e.g. field eqs, initial conditions,
outputs, discrete derivatives…)

Model file: lphi4.h

$$\boxed{\tilde{V}(\tilde{\phi},\tilde{\chi}) = \frac{1}{4}\tilde{\phi}^4 + \frac{1}{2}q\tilde{\phi}^2\tilde{\chi}^2}$$

lphi4.h



```
#ifndef LPHI4_H
#define LPHI4_H

#include "CosmoInterface/cosmointerface.h"

namespace TempLat
{
    // Model name and number of fields
    struct ModelPars : public TempLat::DefaultModelPars {
        static constexpr size_t NScalars = 2;
        static constexpr size_t NPotTerms = 2;
    };

  #define MODELNAME lphi4  // model name

  template<class R>
  using Model = MakeModel(R, ModelPars);

  class MODELNAME : public Model<MODELNAME>
  {
```

Number of scalar fields

$\tilde{\phi}$ field 0

$\tilde{\chi}$ field 1

Number of terms in potential

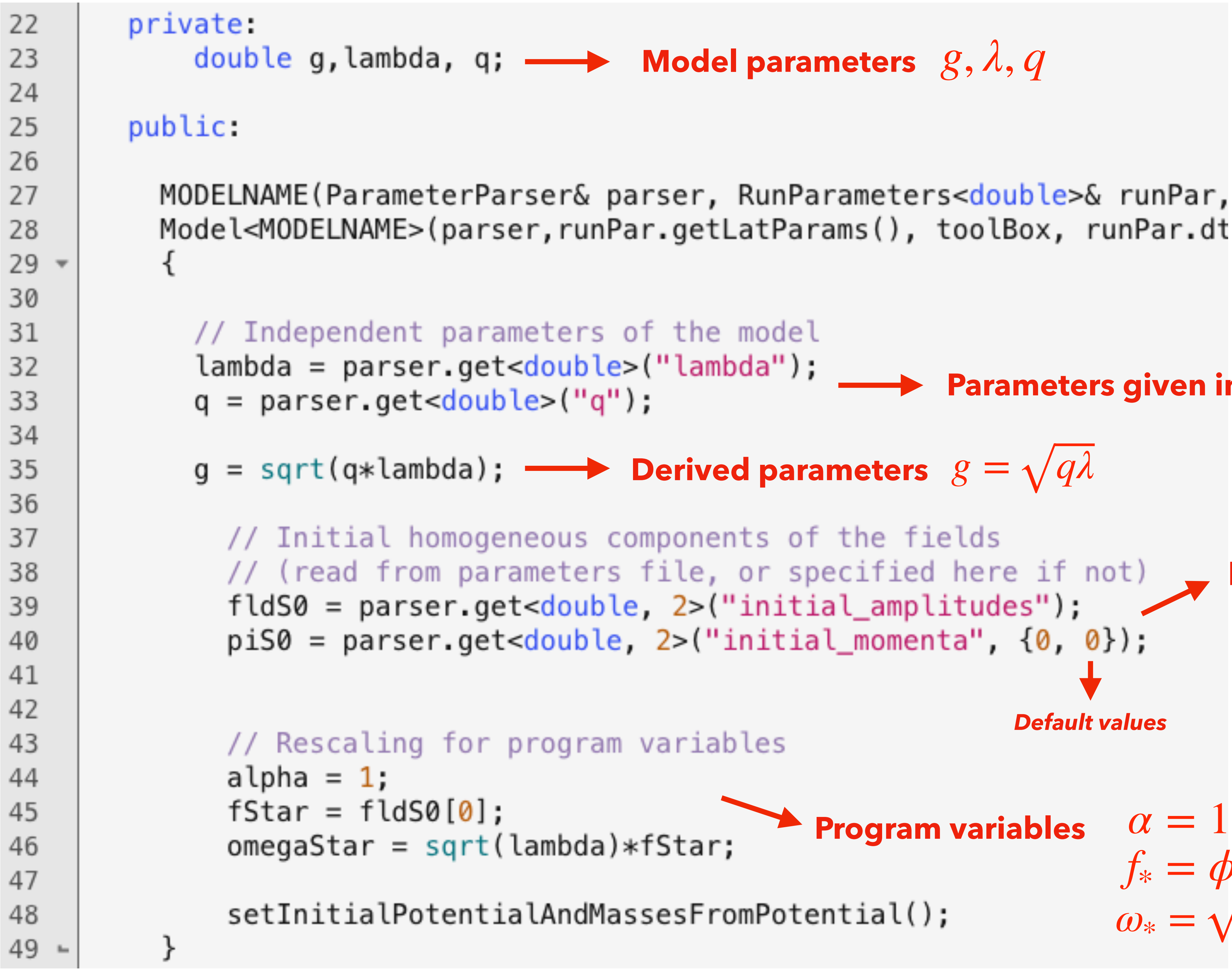
$\tilde{V}^{(0)} = \frac{1}{4}\tilde{\phi}^4$ term 0

$\tilde{V}^{(1)} = \frac{1}{2}q\tilde{\phi}^2\tilde{\chi}^2$ term 1

Model name: it should match the name of the file!

Model file: lphi4.h

lphi4.h



```
  private:
      double g,lambda, q;

  public:

    MODELNAME(ParameterParser& parser, RunParameters<double>& runPar,
    Model<MODELNAME>(parser,runPar.getLatParams(), toolBox, runPar.dt
    {

      // Independent parameters of the model
      lambda = parser.get<double>("lambda");
      q = parser.get<double>("q");

      g = sqrt(q*lambda);

        // Initial homogeneous components of the fields
        // (read from parameters file, or specified here if not)
        fldS0 = parser.get<double, 2>("initial_amplitudes");
        piS0 = parser.get<double, 2>("initial_momenta", {0, 0});


        // Rescaling for program variables
        alpha = 1;
        fStar = fldS0[0];
        omegaStar = sqrt(lambda)*fStar;

        setInitialPotentialAndMassesFromPotential();
    }
```

Model parameters g, λ, q

Parameters given in input file λ, q

Derived parameters $g = \sqrt{q\lambda}$

Vectors of initial homogeneous components
(ϕ_*, χ_*) $(\dot{\phi}_*, \dot{\chi}_*)$

Default values

Program variables

$$\alpha = 1$$
$$f_* = \phi_*$$
$$\omega_* = \sqrt{\lambda}\phi_*$$

Model file: lphi4.h

$$\tilde{V}(\tilde{\phi},\tilde{\chi}) = \frac{1}{4}\tilde{\phi}^4 + \frac{1}{2}q\tilde{\phi}^2\tilde{\chi}^2$$

lphi4.h

```
   // Program potential
    auto potentialTerms(Tag<0>) // term 0
    {
        return 0.25 * pow<4>(fldS(0_c));
    }
    auto potentialTerms(Tag<1>) // term 1
    {
        return 0.5 * q * pow<2>(fldS(0_c) * fldS(1_c));
    }

   // Derivatives of the program potential with respect fields
    auto potDeriv(Tag<0>) // dV/dphi
    {
      return   pow<3>(fldS(0_c)) + q * fldS(0_c) * pow<2>(fldS(1_c)) ;
    }
    auto potDeriv(Tag<1>)  // dV/dchi
    {
      return  q * fldS(1_c) * pow<2>(fldS(0_c));
    }

   //  Second derivatives of the program potential with respect fields
    auto potDeriv2(Tag<0>) // d2V/dphi2
    {
      return  3 * pow<2>(fldS(0_c)) +  q * pow<2>(fldS(1_c)) ;
    }
    auto potDeriv2(Tag<1>) // d2V/dchi2
    {
      return  q * pow<2>(fldS(0_c)) ;
    }
```

Potential terms:

$$\tilde{V}^{(0)} = \frac{1}{4}\tilde{\phi}^4$$

$$\tilde{V}^{(1)} = \frac{1}{2}q\tilde{\phi}^2\tilde{\chi}^2$$

First derivatives:

$$\tilde{V}_{,\tilde{\phi}} = \tilde{\phi}^3 + q\tilde{\chi}^2\tilde{\phi}$$

$$\tilde{V}_{,\tilde{\chi}} = q\tilde{\phi}^2\tilde{\chi}$$

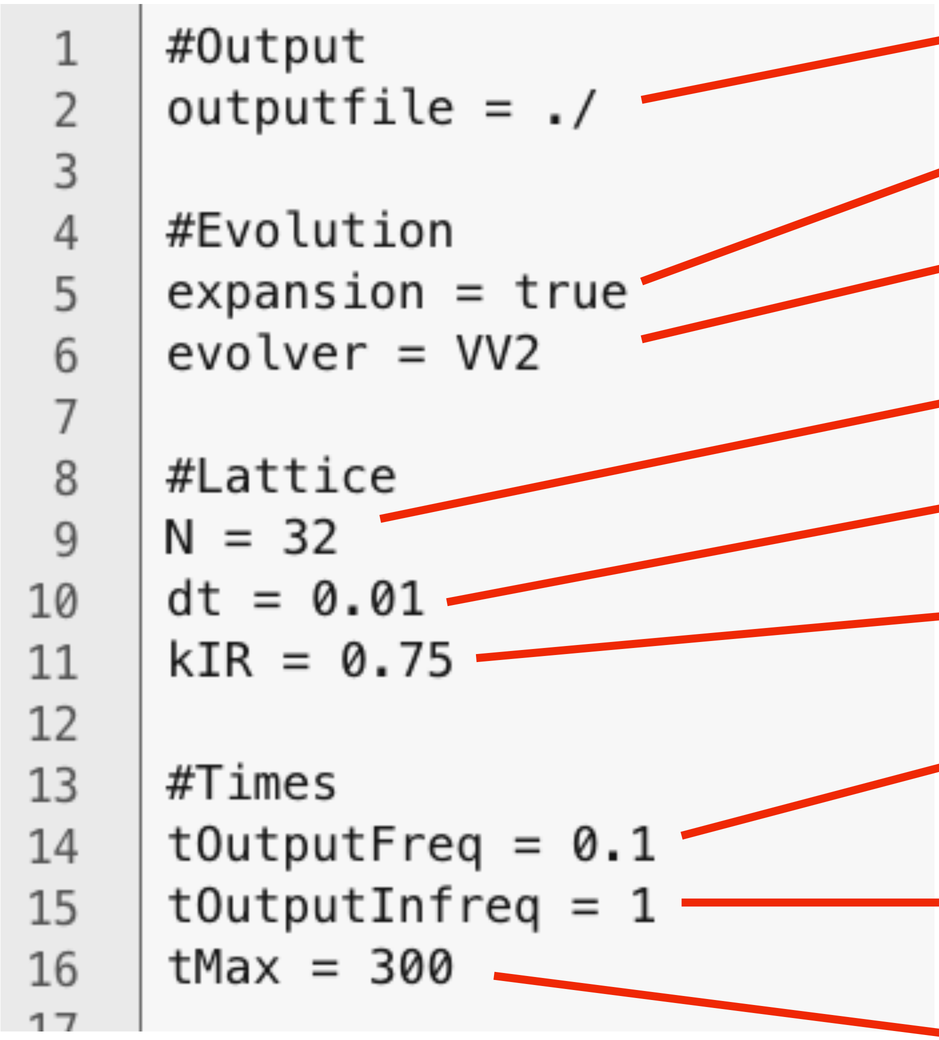
Second derivatives:

$$\tilde{V}_{,\tilde{\phi}\tilde{\phi}} = 3\tilde{\phi}^2 + q\tilde{\chi}^2$$

$$\tilde{V}_{,\tilde{\chi}\tilde{\chi}} = q\tilde{\phi}^2$$

Input file: lphi4.in

lphi4.in



```
#Output
outputfile = ./

#Evolution
expansion = true
evolver = VV2

#Lattice
N = 32
dt = 0.01
kIR = 0.75

#Times
tOutputFreq = 0.1
tOutputInfreq = 1
tMax = 300
```

Output folder: Path where output files are created

(./ : execution folder)

Expansion: true or false

Evolvers: VV2, VV4, VV6, VV8, VV10, LF2

Number of points per dimension: N

Time step: $\delta\tilde{\eta}$

Infrared cutoff of the lattice: $\tilde{k}_{\mathrm{IR}} = k_{\mathrm{IR}}/\omega_*$

Frequency of frequent output
(for volume averages)

Frequency of infrequent output
(for spectra)

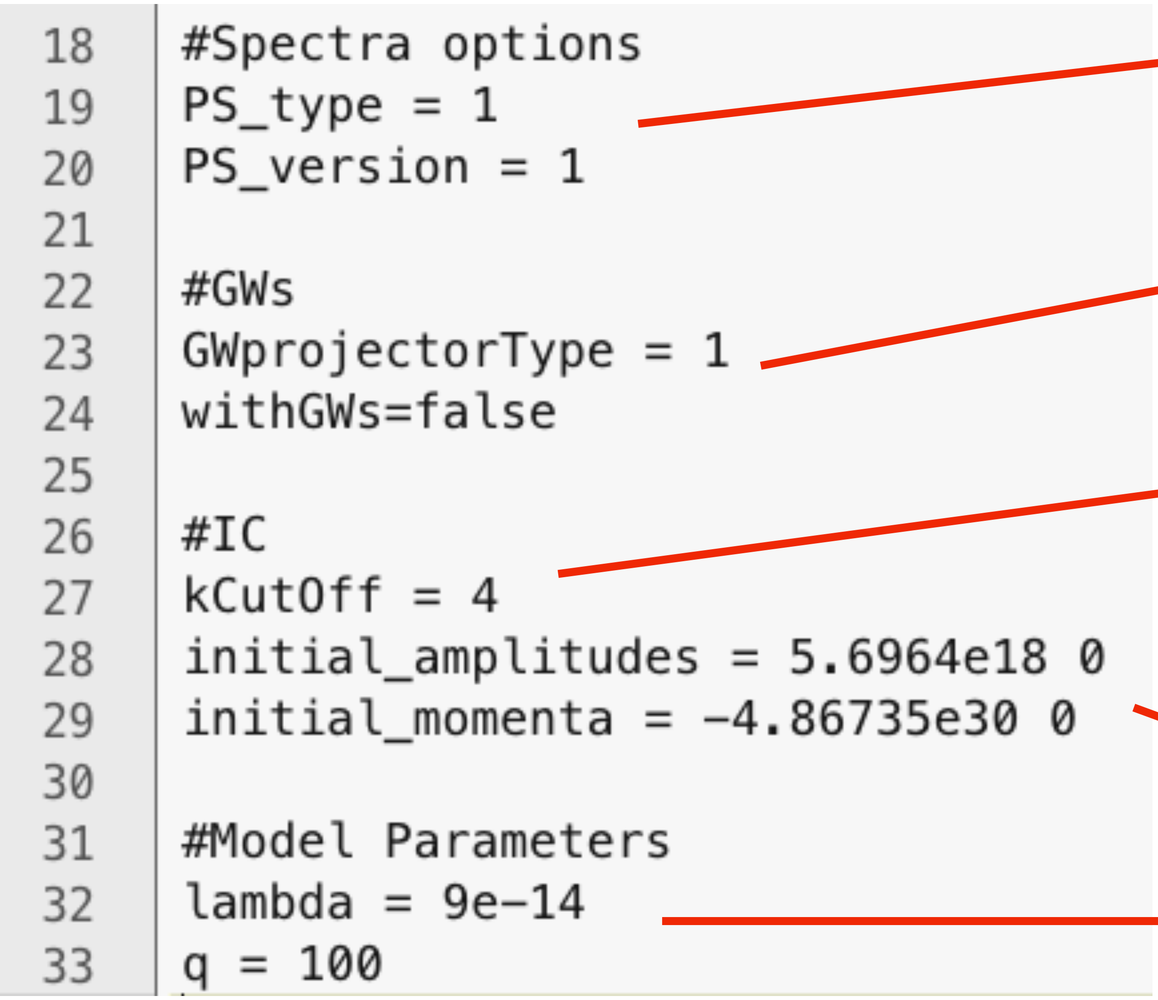
Final time of simulation

in program units!

RULE OF THUMB: If ω_* is set to the oscillation frequency, these numbers should be close to 1

Input file: lphi4.in

lphi4.in



```
#Spectra options
PS_type = 1
PS_version = 1

#GWs
GWprojectorType = 1
withGWs=false

#IC
kCutOff = 4
initial_amplitudes = 5.6964e18 0
initial_momenta = -4.86735e30 0

#Model Parameters
lambda = 9e-14
q = 100
```

Type and version of power spectrum

(See Technical Note I)

Gravitational waves

(See topical lecture)

Cutoff for initial fluctuations

(modes with k> kCutOff are not populated at the initial time)

Initial homogeneous values: for field amplitudes and time-derivatives (**In GeV!!**)

Model parameters: defined in model file

CL: Basic commands

- **Test run (in serial):**

```
cd cosmolattice                    # Enter into main code folder
cd dependencies                    # Enter into dependencies folder
bash fftw3.sh MyFFTW3              # Install FFTw3 from script
cd ..                              # go back to main folder
mkdir build                        # Create a new directory
cd build                           # and go inside it.
cmake -DMODEL=lphi4 ../            # Selects model phi^4 for serial runs
make cosmolattice                  # Compiling
./lphi4 input=../src/models/parameter-files/lphi4.in
                                   # Executes serial run (input
                                     parameter file 'lphi4.in')
```

only if you want to install FFTw3 from the provided script

- **To compile a second model (e.g. tanh2.h):**

```
rm CMakeCache.txt
cmake - DMODEL=tanh2 ../
make cosmolattice
```

Output produced by CL (scalar model)

- **[model_name].infos:** Information about the run.
- **average_scale_factor.txt:** $\tilde{\eta}, a, a', a'/a$
- **average_scalar_[n].txt:** $\tilde{\eta}, \langle \tilde{\phi}_n \rangle, \langle \tilde{\phi}'_n \rangle, \langle \tilde{\phi}^2_n \rangle, \langle \tilde{\phi}'^2_n \rangle, rms(\tilde{\phi}_n), rms(\tilde{\phi}'_n)$
- **average_energy_conservation.txt:** $\tilde{\eta}, \dfrac{\langle LHS - RHS \rangle}{\langle LHS + RHS \rangle}, \langle LHS \rangle, \langle RHS \rangle$

$$a'^2 = \frac{a^{2\alpha+2}}{3}\left(\frac{f_*}{m_p}\right)^2 \tilde{\rho}$$

LHS

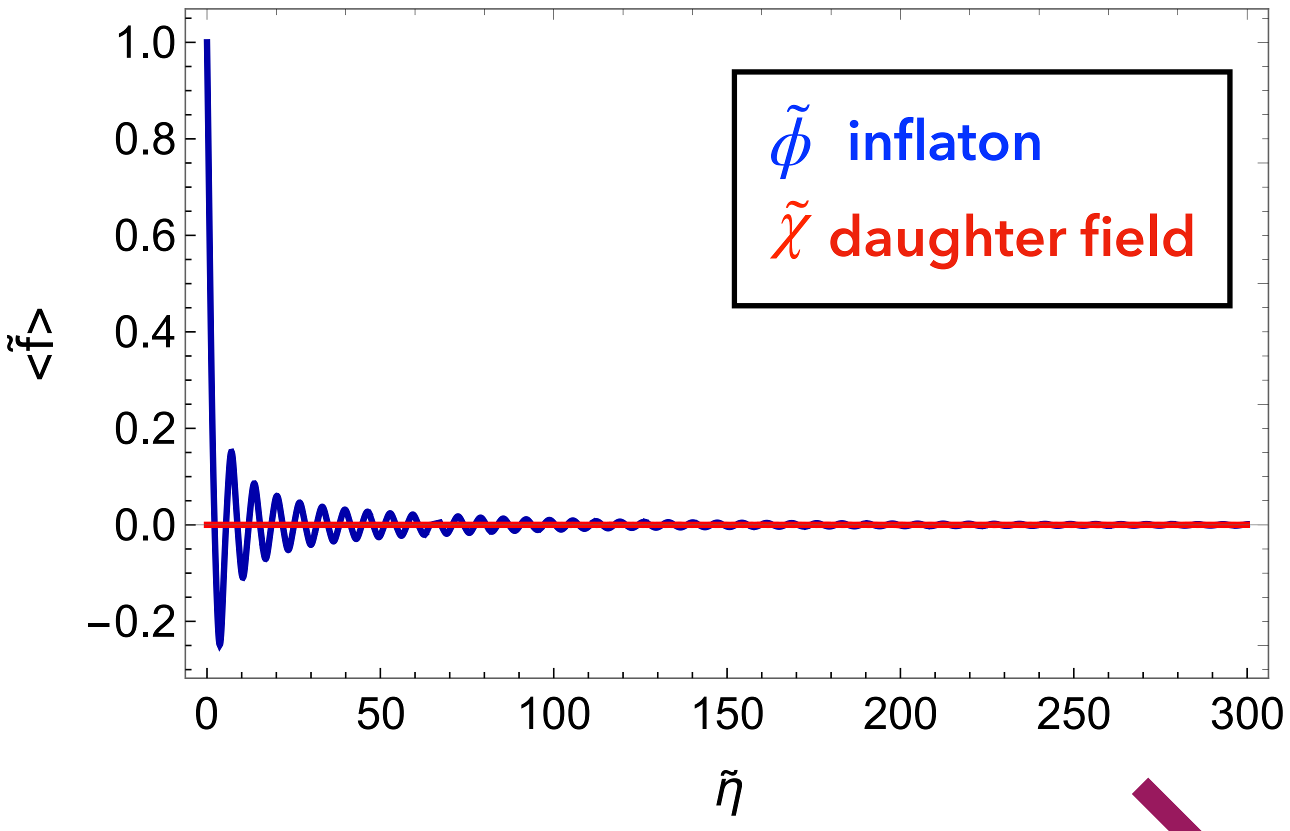
RHS

- **average_energies.txt:** $\tilde{\eta}, \tilde{E}^{(0)}_K, \tilde{E}^{(0)}_G, \ldots, \tilde{E}^{(N_s-1)}_K, \tilde{E}^{(N_s-1)}_G, \tilde{E}^{(0)}_V, \ldots, \tilde{E}^{(N_p-1)}_V, \langle \tilde{\rho} \rangle$
- **spectra_scalar_[n].txt:** $\tilde{k}, \widetilde{\Delta}_{\tilde{\phi}}(\tilde{k}), \widetilde{\Delta}_{\tilde{\phi}'}(\tilde{k}), \tilde{n}_{\tilde{k}}, \Delta n_{bin}$

$$\widetilde{\Delta}_{\tilde{\phi}} \equiv \frac{\Delta_\phi}{f_*^2} \qquad \widetilde{\Delta}_{\tilde{\phi}'} \equiv \frac{\Delta_{\phi'}}{f_*^2 \omega_*^2}$$

Table of contents

Output: field amplitudes



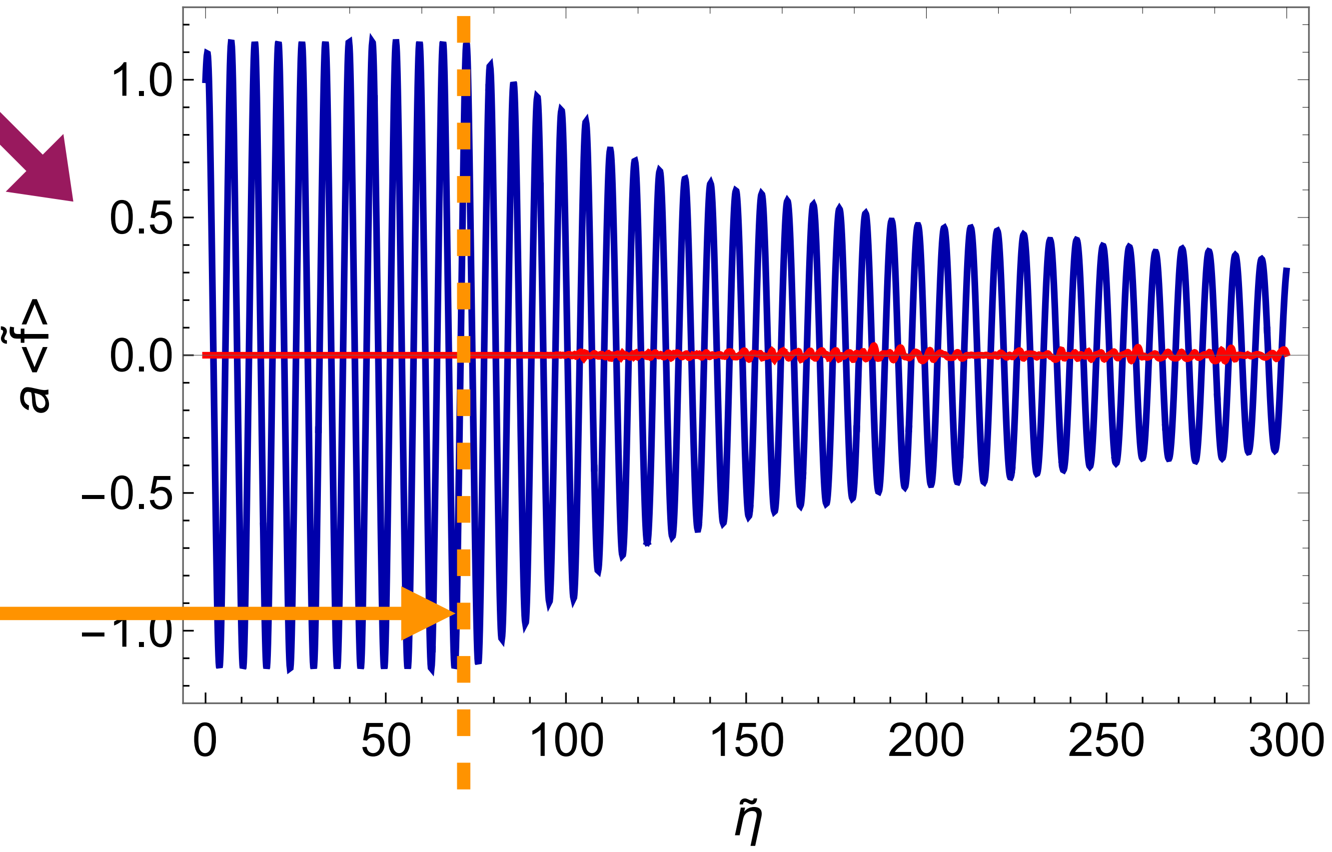
Lattice parameters:

$N = 32$
$\delta\tilde{\eta} = 0.01$
$\tilde{k}_{\rm IR} = 0.75$

Model parameters:

$q \equiv g^2/\lambda = 100$
$\lambda = 9 \cdot 10^{-14}$

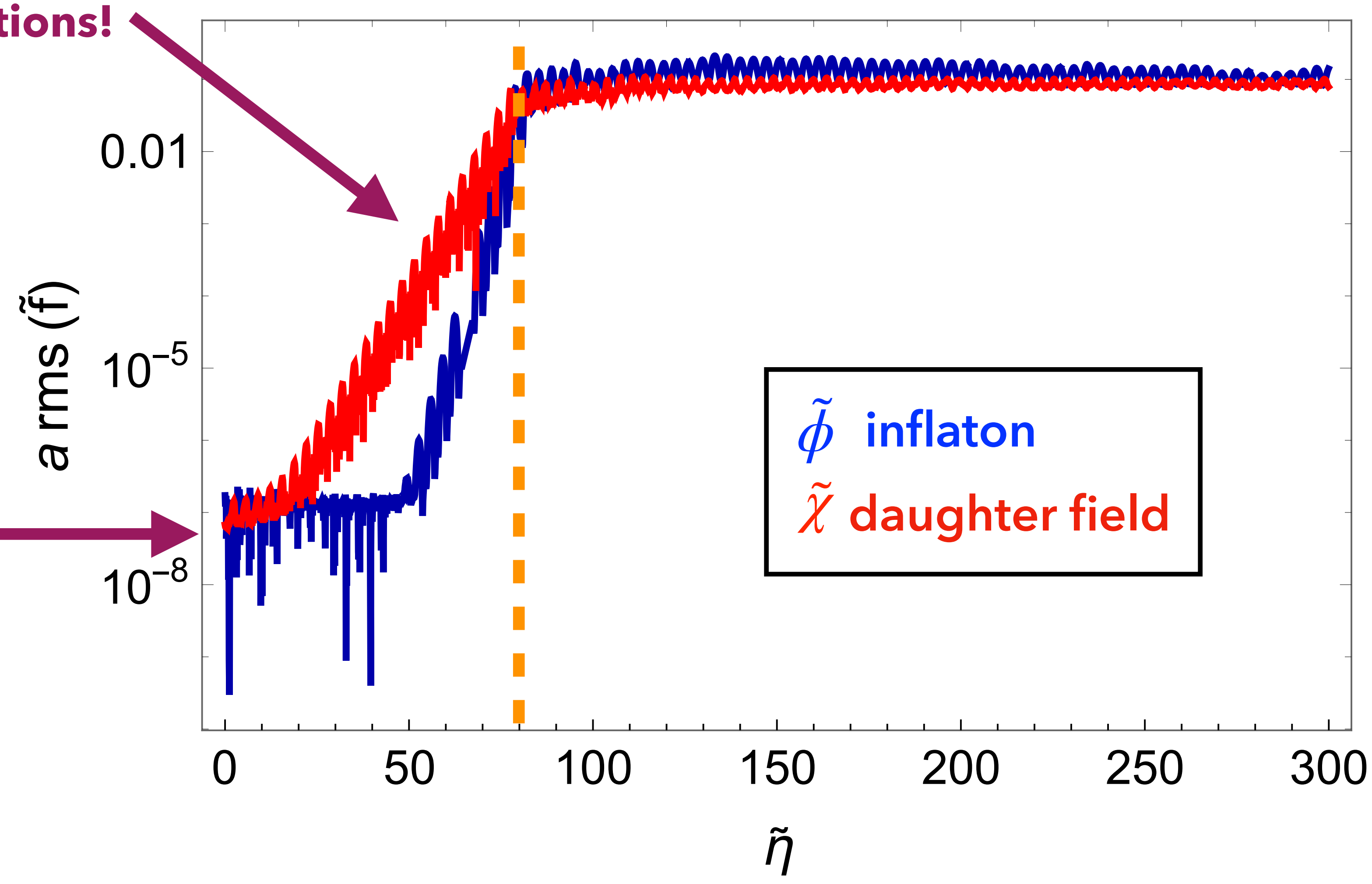
conformal transformation



start of non-linear regime

Output: variances

$$rms(\tilde{f}) \equiv \sqrt{\langle \tilde{f}^2 \rangle - \langle \tilde{f} \rangle^2} \qquad \tilde{f} = \tilde{\varphi}, \tilde{\chi}$$

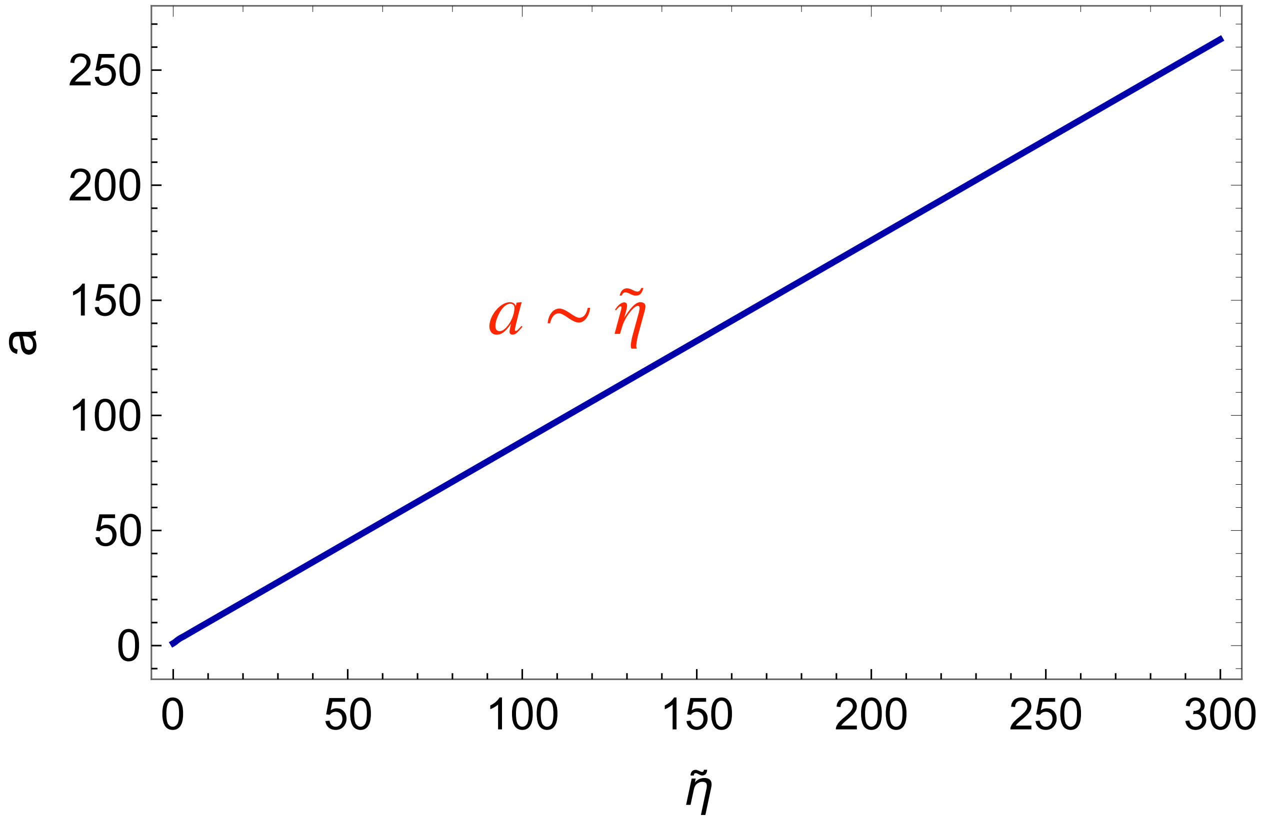


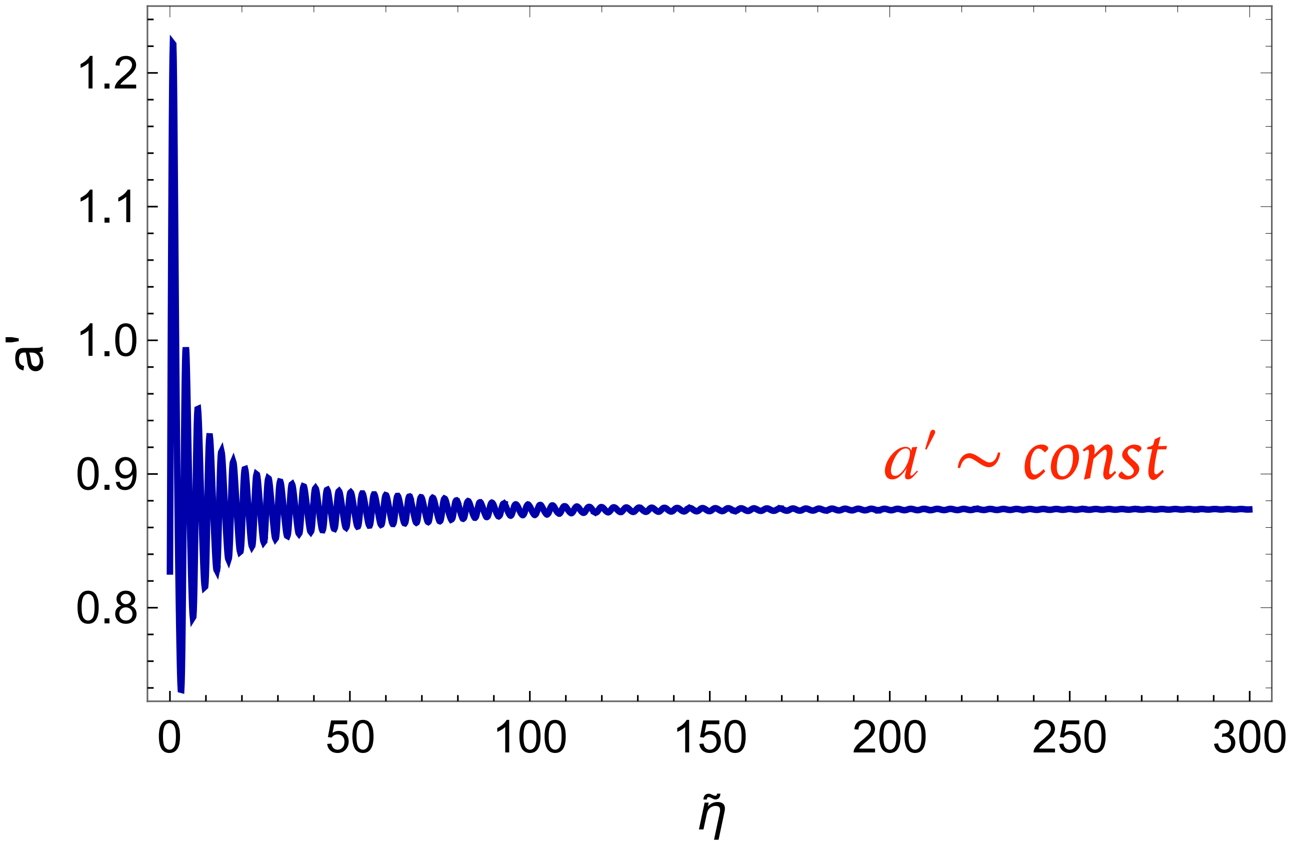
Exponential growth of fluctuations!

(initial) vacuum fluctuations

The excitation of the daughter field during the linear regime is stronger than for the inflaton, and dominates the energy budget

Output: scale factor

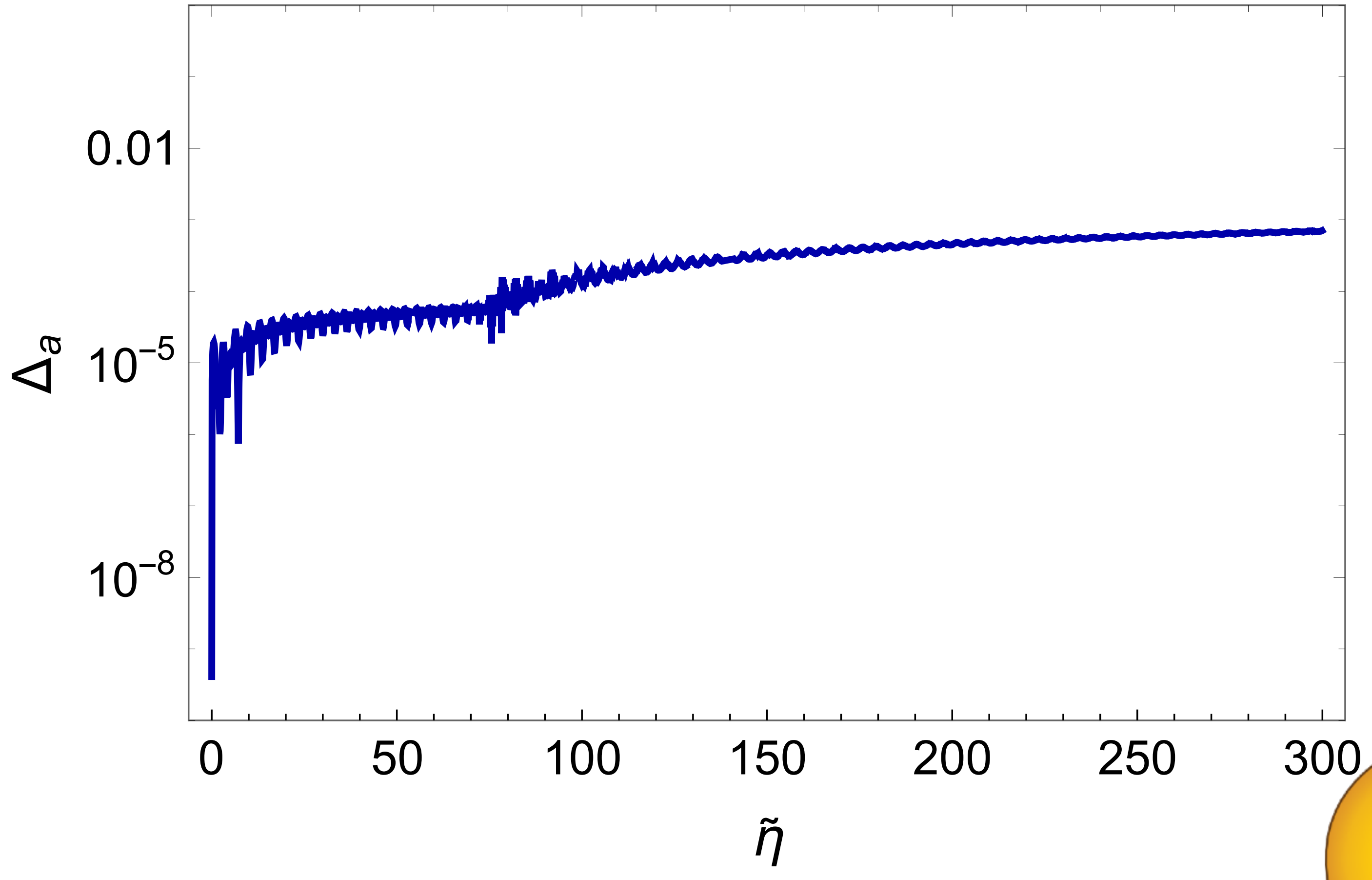




The scale factor behaves as radiation-dominated during the entire post-inflationary history

Output: energy conservation

- Algorithms use **second Friedmann equation** to **evolve the scale factor**.
- The **first Friedmann equation** can be used to check the accuracy of the simulation. We denote this “energy conservation”.

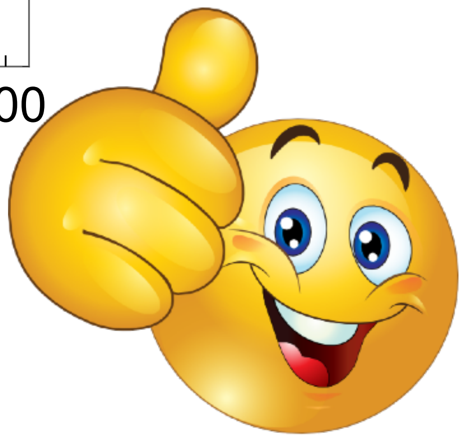


$$a'^2 = \frac{a^{2\alpha+2}}{3}\left(\frac{f_*}{m_p}\right)^2 \tilde{\rho}$$

$$\Downarrow$$

$$\Delta_a \equiv \frac{\langle \mathrm{LHS} - \mathrm{RHS} \rangle}{\langle \mathrm{LHS} + \mathrm{RHS} \rangle}$$

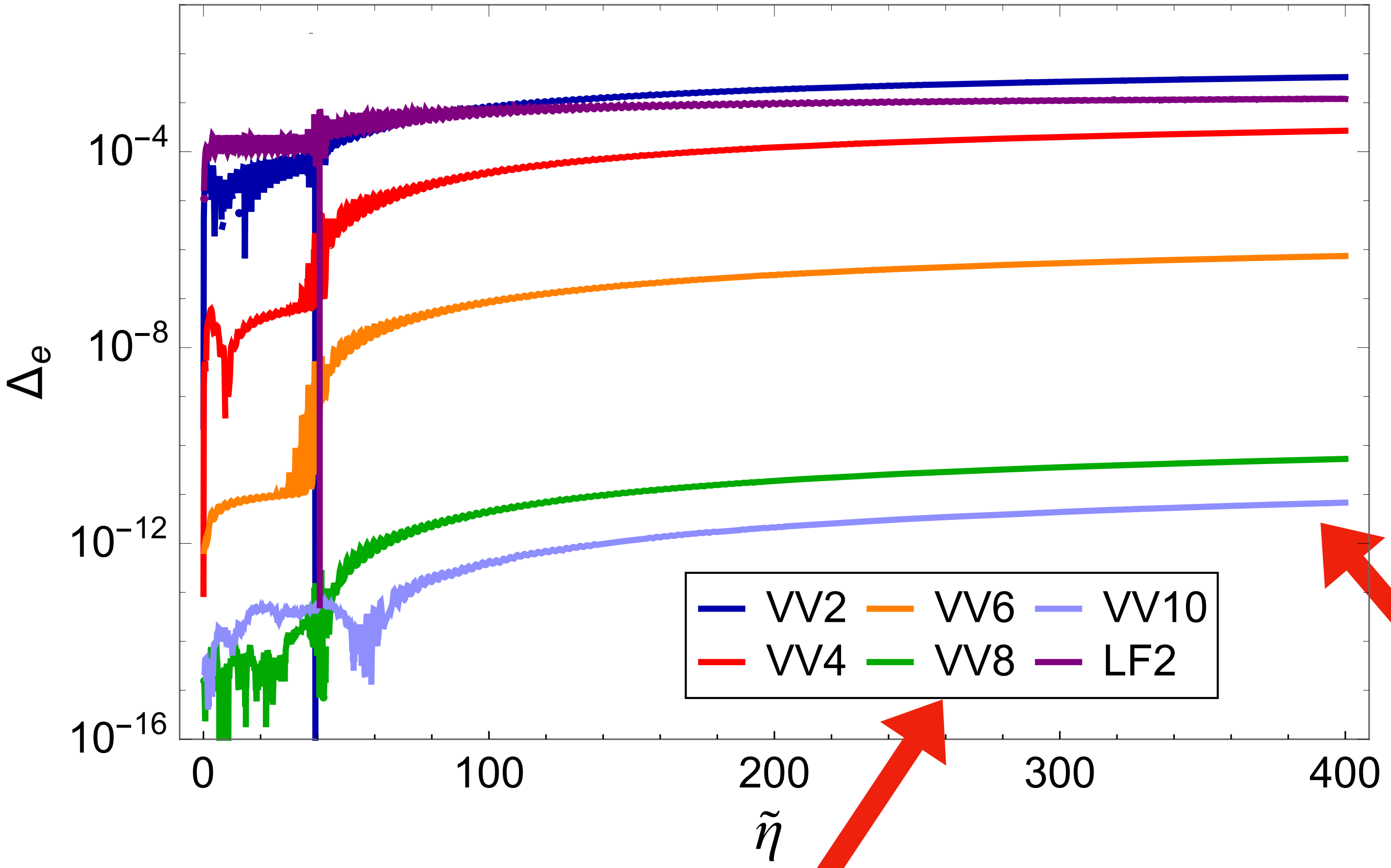
(we must have $\Delta_a \ll 1$)



GOOD
ENERGY
CONSERVATION!

Disgression: energy conservation and accuracy of evolver

- Energy conservation can be improved if we increase the order of accuracy of the simulation:



Evolution algorithms:

- **VVN:** Velocity-verlet of accuracy order O(dt^n)
- **LF2:** Staggered leapfrog, accuracy order O(dt^2)

Energy conserved up to machine precision for VV10!

Output: energy components

$$\langle\tilde{\rho}\rangle \equiv \frac{\langle\rho\rangle}{f_*^2\omega_*^2} = \left(\tilde{E}_K^{(0)} + \tilde{E}_V^{(0)} + \tilde{E}_G^{(0)} + \tilde{E}_K^{(1)} + \tilde{E}_G^{(1)} + \tilde{E}_V^{(1)}\right)$$

Kinetic inflaton

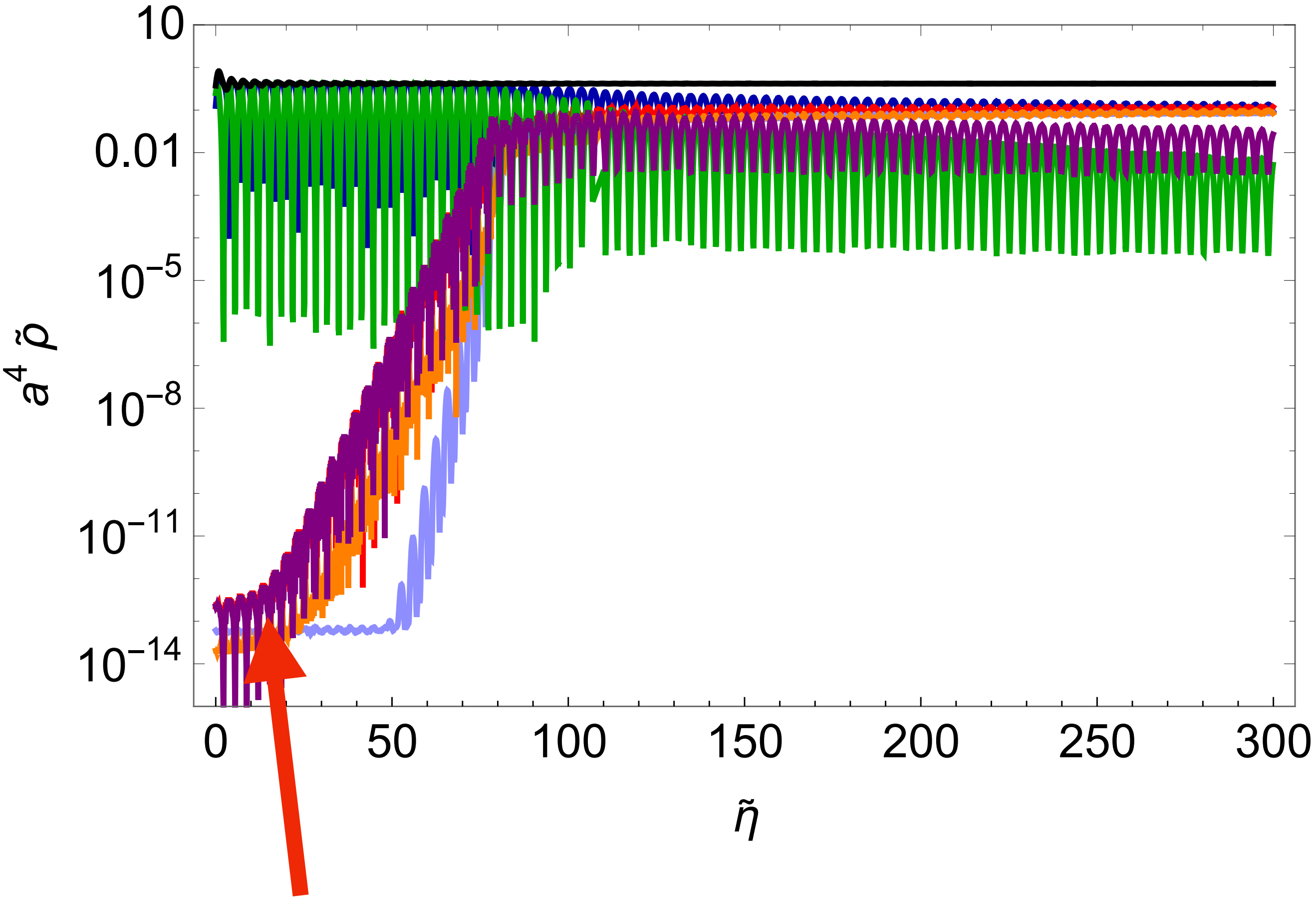
Potential inflaton

Gradient inflaton

Kinetic daughter

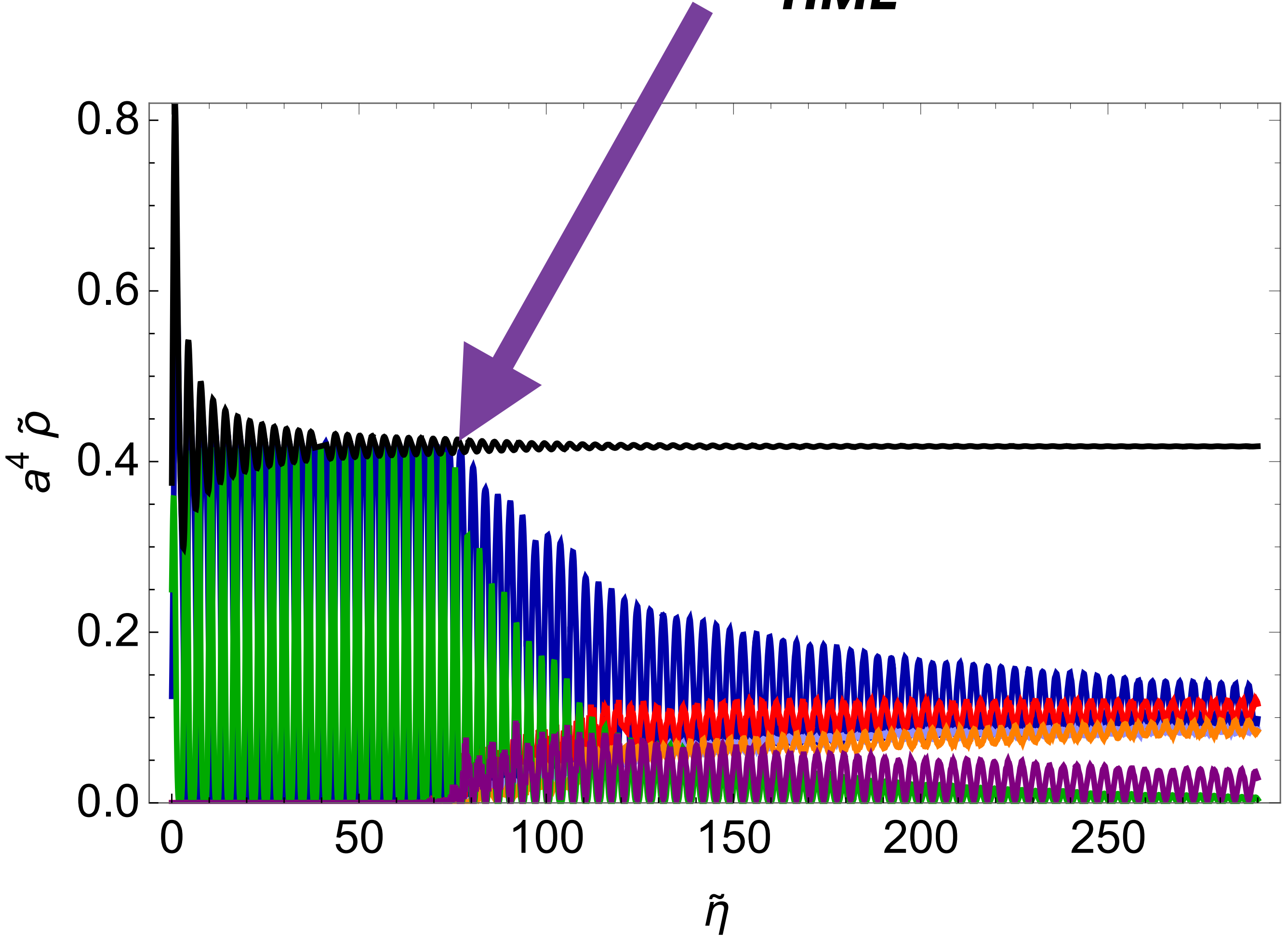
Gradient daughter

Interaction energy

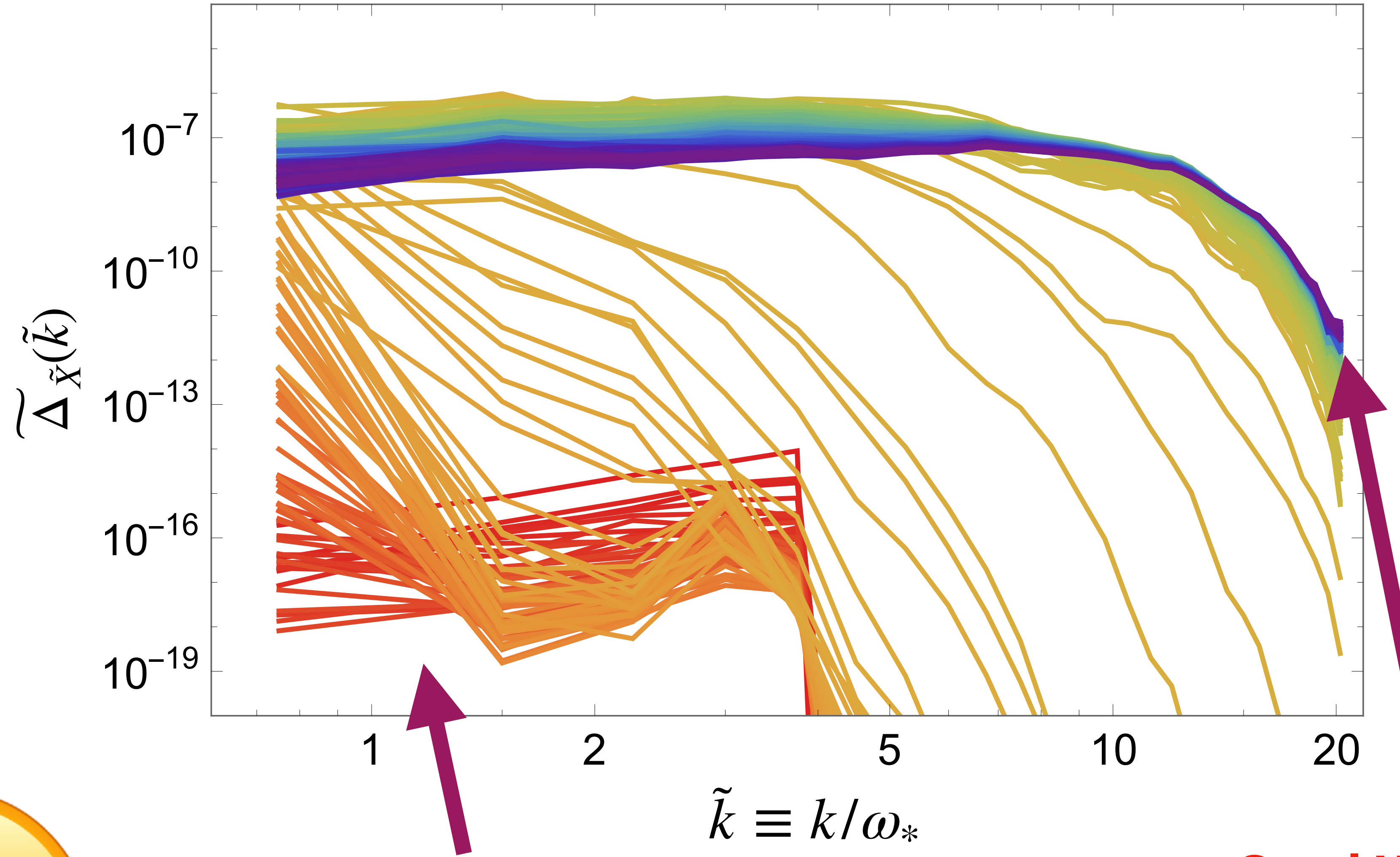


EXPONENTIAL GROWTH

"BACKREACTION" TIME



Output: field spectra



BAD IR COVERAGE!

Dominant resonance band not well captured

Good UV coverage:

UV tail (generated during the non-linear regime) well captured

Lattice parameters and IR/UV coverage

- **Infrared cutoff:** $\tilde{k}_{\rm IR} = \frac{2\pi}{\tilde{L}} \overset{\tilde{L} = N\delta\tilde{x}}{=} \frac{2\pi}{N\delta\tilde{x}}$
- **Ultraviolet cutoff:** $\tilde{k}_{\rm max} = \sqrt{3}\frac{N}{2}\tilde{k}_{\rm IR} = \frac{\pi}{\delta\tilde{x}}$

- ***In our simulation:***

$N = 32 \quad \tilde{k}_{\rm IR} = 0.75 \quad \delta\tilde{\eta} = 0.01$

$\downarrow$

$\tilde{k}_{\rm max} = 20.8 \quad \delta\tilde{x} = 0.26$

We want to decrease $\tilde{k}_{\rm IR}$. There are two ways:

1) Increase N (but increases simulation time as ~N^3!)

2) Increase $\delta\tilde{x}$ (but reduces $\tilde{k}_{\rm UV}$)

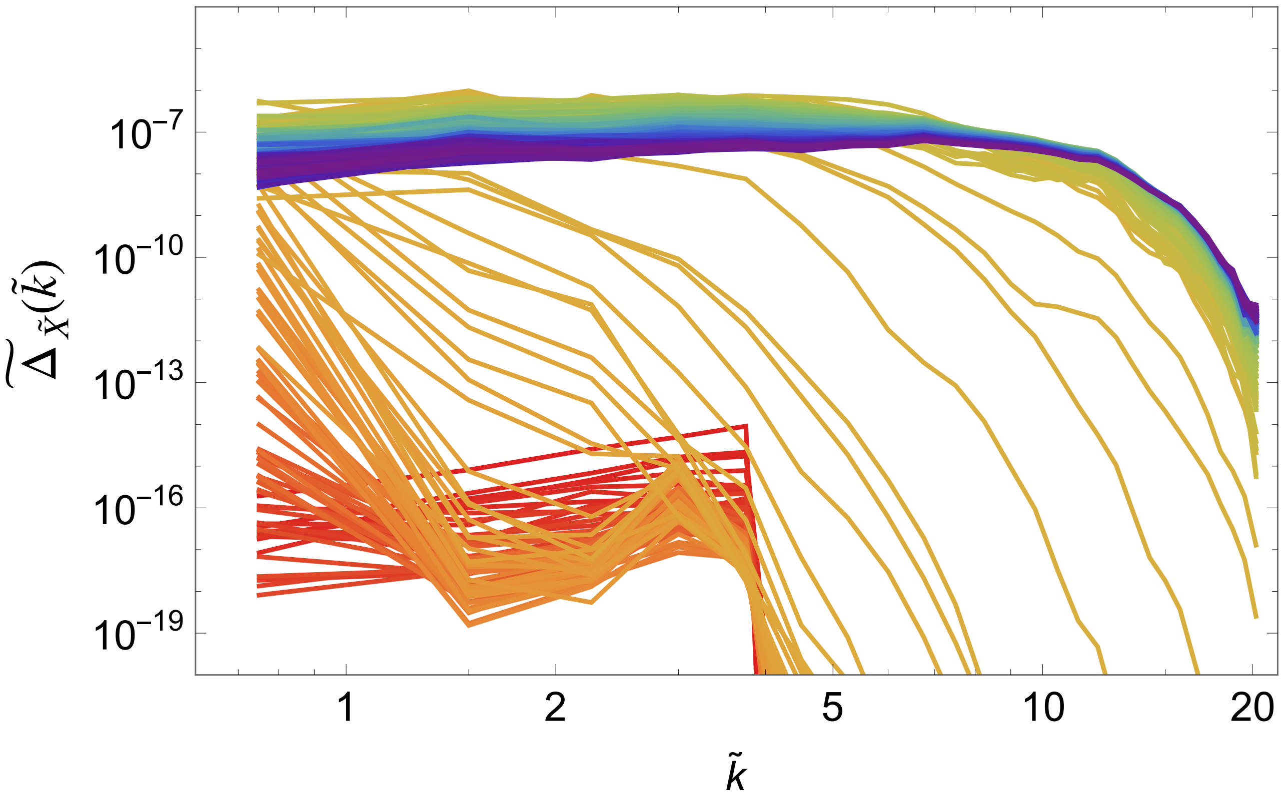
↳ Time step must also be reduced to preserve the Courant stability condition:

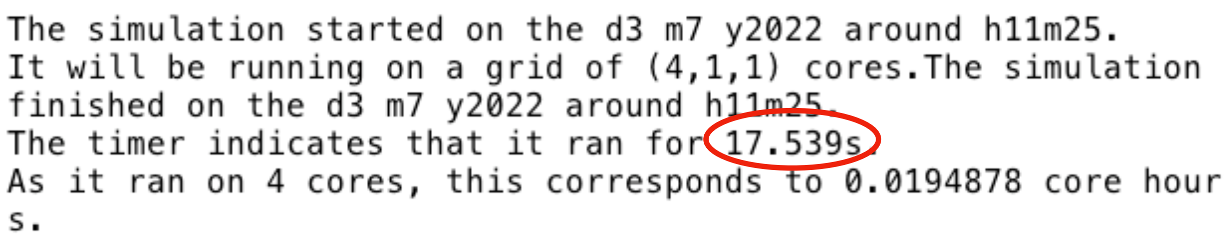
$$\frac{\delta\tilde{\eta}}{\delta\tilde{x}} < \frac{1}{\sqrt{N_{\rm dims}}}$$

Lattice parameters and IR/UV coverage

1) Increase N:

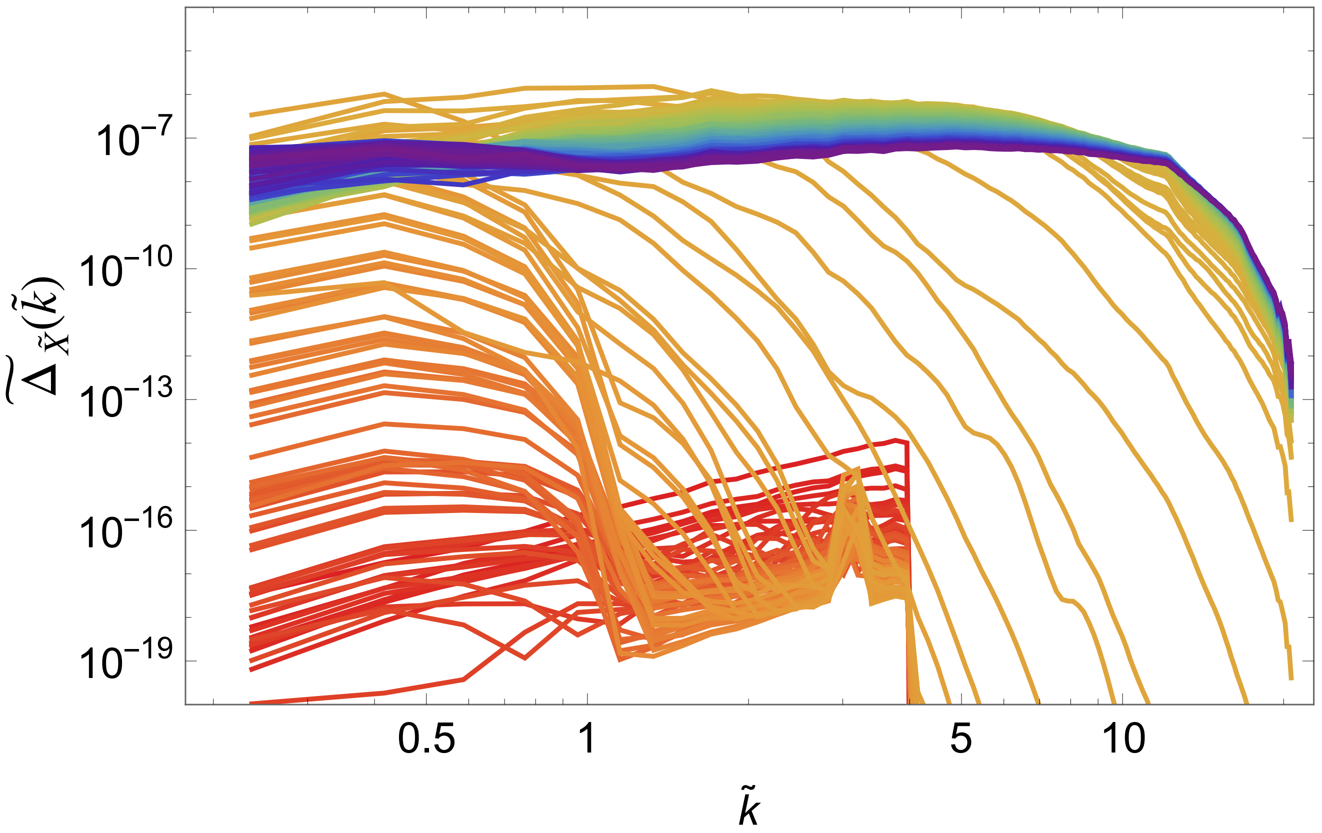
$N = 32$ $\delta\tilde{x} = 0.26$ → $\tilde{k}_{\rm IR} = 0.75$

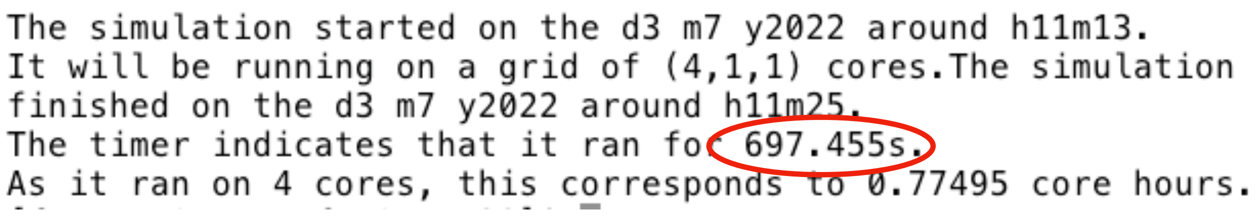




```
The simulation started on the d3 m7 y2022 around h11m25.
It will be running on a grid of (4,1,1) cores.The simulation
finished on the d3 m7 y2022 around h11m25.
The timer indicates that it ran for 17.539s.
As it ran on 4 cores, this corresponds to 0.0194878 core hour
s.
```

$N = 128$ $\delta\tilde{x} = 0.26$ → $\tilde{k}_{\rm IR} = 0.188$



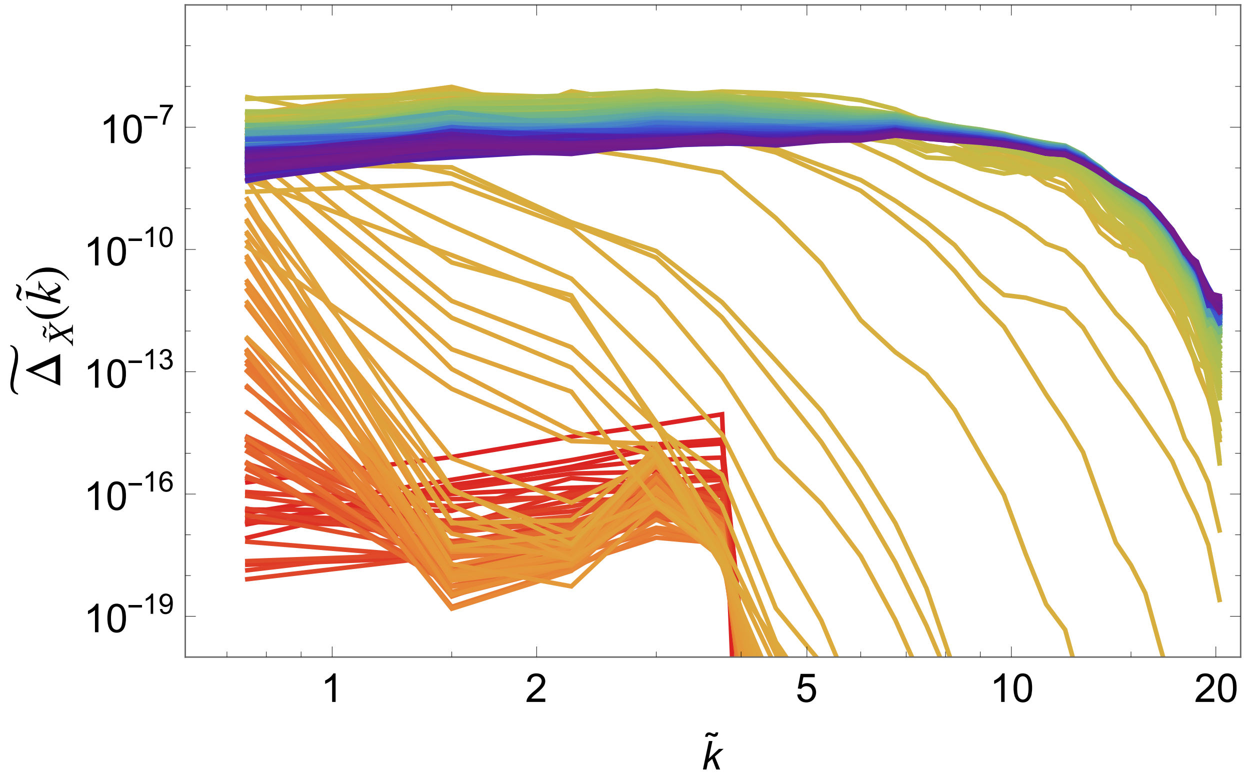


```
The simulation started on the d3 m7 y2022 around h11m13.
It will be running on a grid of (4,1,1) cores.The simulation
finished on the d3 m7 y2022 around h11m25.
The timer indicates that it ran for 697.455s.
As it ran on 4 cores, this corresponds to 0.77495 core hours.
```

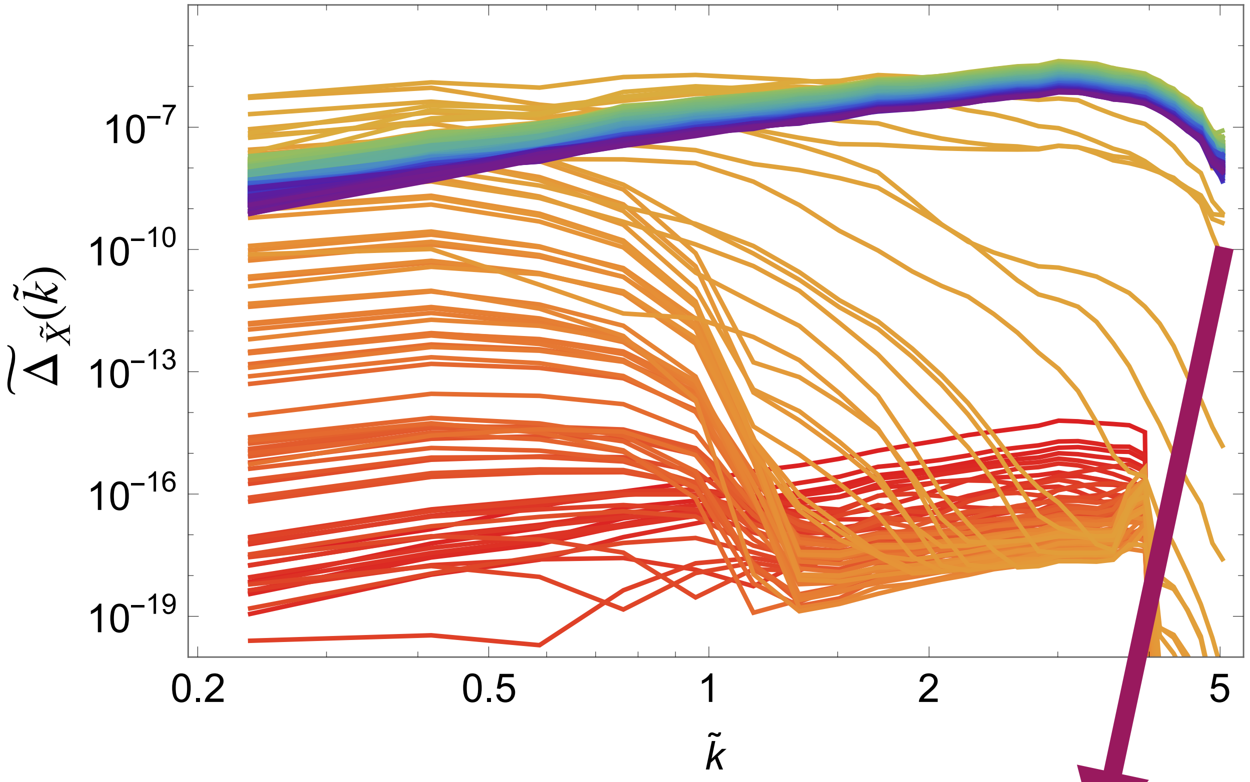
Lattice parameters and IR/UV coverage

2) Increase $\delta\tilde{x}$:

$N = 32$ $\quad \delta\tilde{\eta} = 0.01$ $\quad \delta\tilde{x} = 0.26$ $\xrightarrow{}$ $\tilde{k}_{\rm IR} = 0.75$ $\quad \tilde{k}_{\rm max} = 20.8$



$N = 32$ $\quad \delta\tilde{\eta} = 0.0025$ $\quad \delta\tilde{x} = 1.04$ $\xrightarrow{}$ $\tilde{k}_{\rm IR} = 0.188$ $\quad \tilde{k}_{\rm max} = 5.2$



BAD UV COVERAGE!

Table of contents

Proposed exercises

1. Run the **lphi4** model with the default parameters (the ones provided in the lphi4.in input file). Plot figures for the volume-averaged field amplitudes and variances, energy components, and field spectra. You can use the provided mathematica or python notebooks.

2. Run the **lphi4** model with the same parameters as in exercise 1, but changing the resonance parameter to i) $q \equiv g^2/\lambda = 5$; and ii) $q=0$. Plot the field spectra and observe the differences with respect to exercise 1 (where $q=100$).

3. Modify the **lphi4** model to add a second daughter field coupled to the inflaton:

$$V(\phi) = \frac{1}{4}\lambda\phi^4 + \frac{1}{2}g_1^2\phi^2X_1^2 + \frac{1}{2}g_2^2\phi^2X_2^2$$

Run the model for parameters: $q_1 \equiv \frac{g_1^2}{\lambda} = 10^2$ $\quad q_2 \equiv \frac{g_2^2}{\lambda} = 10^2$ $\quad \lambda = 9 \cdot 10^{-14}$

Proposed exercises

4. Run the **lphi4** model with the default parameters (lphi4.in), but:
   a) Change the evolution algorithm from VV2 to VV4.
   b) Reduce the time step by half.

   How does the energy conservation improve in each of the two cases with respect to the simulation of exercise 1. Which of the two simulations is faster? Note: The simulation time is given in the info file.

5. Run the **tanh2** model provided with the code, which implements the following potential:

$$V(\phi) = \frac{\Lambda^4}{2}\tanh^2\left(\frac{\phi}{M}\right) + \frac{1}{2}g^2\phi^2X^2$$

Thank you!