第三届高能物理理论与实验融合发展研讨会

PhaseTracer2 软件

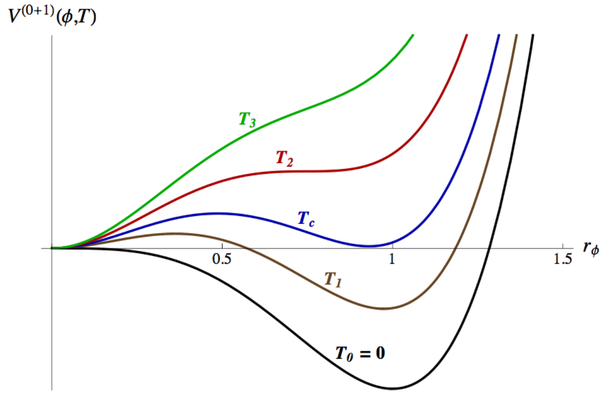
张阳　河南师范大学

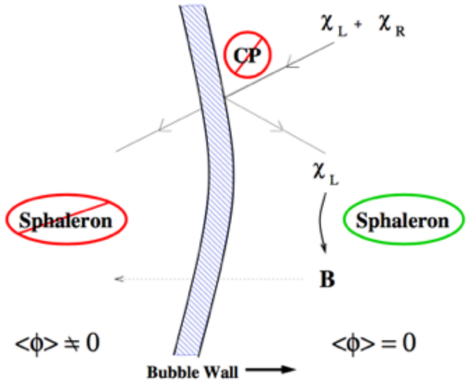
Based on https://github.com/PhaseTracer and arXiv:2411.xxxxx

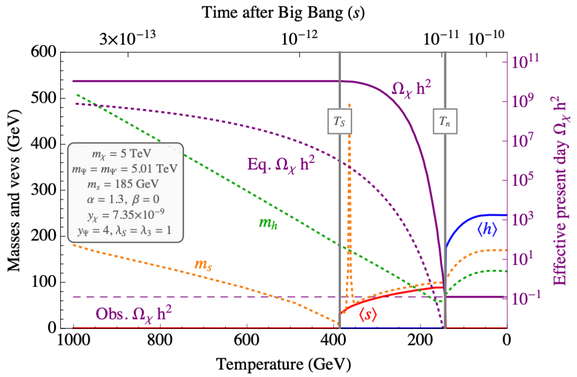
In collaboration with Peter Athron, Csaba Balázs, Andrew Fowlie, William Searle, Yang Xiao

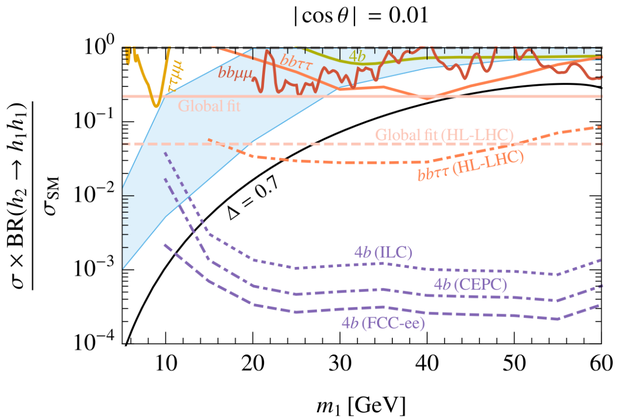
辽宁师范大学　　2024.11.01

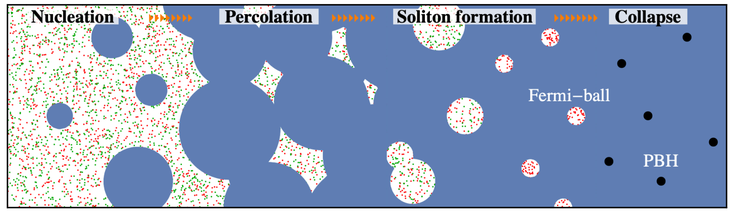
Introduction



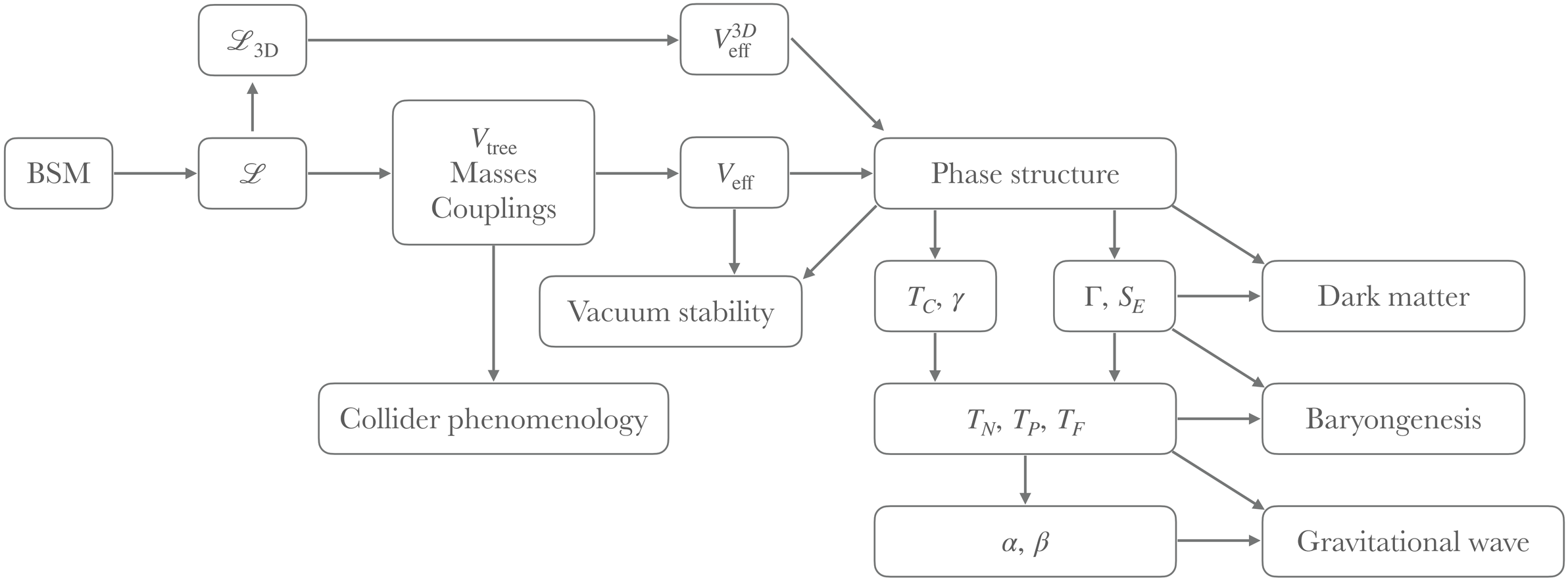




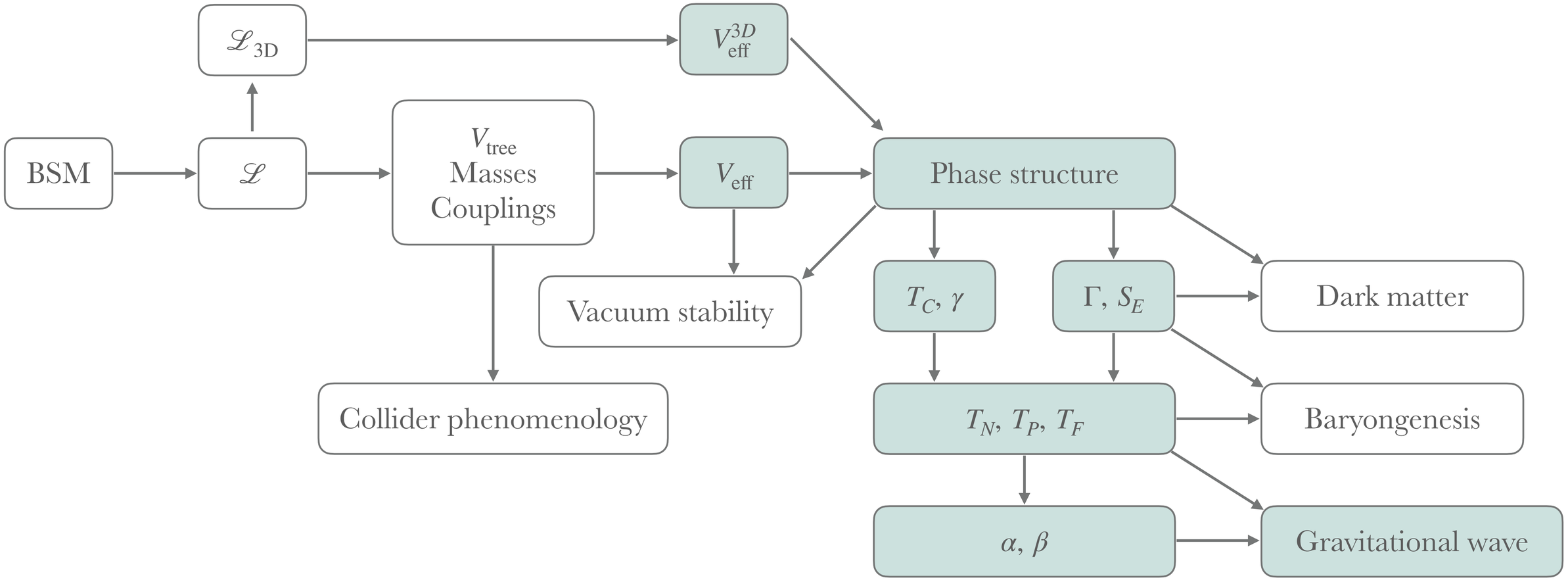




Introduction



Introduction



PhaseTracer2

FlexibleSUSY

DRalgo

Input: Effective potential

- $\overline{\text{MS}}$ Scheme
- HT Expansion
- OS-like Scheme
- 3D EFT
- R_ξ Gauge
- Covariant Gauge
- Arnold-Espinosa Approach
- Parwani Approach

BubbleProfiler

TransitionSolver

Outputs

- Phase structure
- Critical Temperature
- Bounce Action
- Nucleation Temperature
- Strength factor
- Transition duration
- Gravitational Wave Spectrum
- Gravitational Wave SNR

Example

- Model: SM, Singlet extension, 2HDM, NMSSM
- Input: V_{tree}, num of fields, symmetry, field-dependent masses, Debye masses

```cpp
#include "one_loop_potential.hpp"
#include "pow.hpp"
#include "SM_parameters.hpp"


namespace EffectivePotential {

class xSM_base : public OneLoopPotential {
 public:
  double V0(Eigen::VectorXd phi) const override {
    return 0.5 * muh_sq * square(phi[0]) +
           0.25 * lambda_h * pow_4(phi[0]) +
           0.25 * lambda_hs * square(phi[0]) * square(phi[1]) +
           0.5 * mus_sq * square(phi[1]) +
           0.25 * lambda_s * pow_4(phi[1]);
  }
  size_t get_n_scalars() const override {
    return 2;
  }
  std::vector<Eigen::VectorXd> apply_symmetry(Eigen::VectorXd phi) const override {
    auto phi1 = phi;
    phi1[0] = - phi[0];
    auto phi2 = phi;
    phi2[1] = - phi[1];
    return {phi1, phi2};
  };
```

$$V_0(h,s) = \frac{\mu_H^2}{2}h^2 + \frac{\lambda_H}{4}h^4 + \frac{\mu_S^2}{2}s^2 + \frac{\lambda_S}{4}s^4 + \frac{\lambda_{HS}}{4}h^2s^2$$

$$n_{\text{scalars}} = 2$$

$$V(h,s) = V(-h,s), V(h,s) = V(h,-s)$$

Example

- Model: SM, Singlet extension, 2HDM, NMSSM
- Input: V_{tree}, num of fields, symmetry, field-dependent masses, Debye masses

```
58  std::vector<double> get_fermion_masses_sq(Eigen::VectorXd phi) const override {
59    return {0.5 * SM_yt_sq * square(phi[0]),
60            0.5 * SM_yb_sq * square(phi[0]),
61            0.5 * SM_ytau_sq * square(phi[0])};
62  }
63  std::vector<double> get_fermion_dofs() const override {
64    return {12., 12., 4.};
65  }
66  std::vector<double> get_scalar_thermal_sq(double T) const override {
67    const double c_h = (9. * square(SM_g) +
68                        3. * square(SM_gp) +
69                        2. * (6. * SM_yt_sq + 6. * SM_yb_sq +
70                              2. * SM_ytau_sq + 12. * lambda_h + lambda_hs)) / 48.;
71    const double c_s = (2. * lambda_hs + 3. * lambda_s) / 12.;
72    return {c_h * square(T), c_s * square(T)};
73  }
74  std::vector<double> get_scalar_debye_sq(Eigen::VectorXd phi, double xi, double T) const override{
75    const double h = phi[0];
76    const double s = phi[1];
77    const auto thermal_sq = get_scalar_thermal_sq(T);
78
79    const double mhh2 = muh_sq + 3. * lambda_h * square(h) + 0.5 * lambda_hs * square(s);
80    ... ...
81  }
82  std::vector<double> get_scalar_dofs() const override {
83    return {1., 1., 1., 1., 1};
84  }
```

$$m^2_{b,t,\tau}(\phi) = \frac{y^2_{b,t,\tau}}{2}\phi_h^2$$

$$n_t = 12,\ n_b = 12,\ n_\tau = 4$$

$$\overline{m}_h^2(\phi,T) = -\left(\mu_h^2+\mu_s^2\right) + 3\left(\lambda_h\phi_h^2+\lambda_s\phi_s^2\right) + \frac{1}{2}\lambda_{hs}\left(\phi_h^2+\phi_s^2\right) + \left(c_h+c_s\right)T^2 + \left[\left(-\left(\mu_h^2-\mu_s^2\right) + 3\left(\lambda_h\phi_h^2-\lambda_s\phi_s^2\right) - \frac{1}{2}\lambda_{hs}\left(\phi_h^2-\phi_s^2\right) + \left(c_h-c_s\right)T^2\right)^2 + 4\lambda_{hs}^2\phi_h^2\phi_s^2\right]^{\frac{1}{2}}, \tag{50}$$

$$\overline{m}_s^2(\phi,T) = -\left(\mu_h^2+\mu_s^2\right) + 3\left(\lambda_h\phi_h^2+\lambda_s\phi_s^2\right) + \frac{1}{2}\lambda_{hs}\left(\phi_h^2+\phi_s^2\right) + \left(c_h+c_s\right)T^2 - \left[\left(-\left(\mu_h^2-\mu_s^2\right) + 3\left(\lambda_h\phi_h^2-\lambda_s\phi_s^2\right) - \frac{1}{2}\lambda_{hs}\left(\phi_h^2-\phi_s^2\right) + \left(c_h-c_s\right)T^2\right)^2 + 4\lambda_{hs}^2\phi_h^2\phi_s^2\right]^{\frac{1}{2}}, \tag{51}$$

$$m_G^2(\phi,T) = \mu_h^2 + \lambda_h\phi_h^2 + \frac{1}{2}\lambda_{hs}\phi_s^2 + c_h T^2, \tag{52}$$

$$n_h = 1,\ n_s = 1,\ n_G = 3$$

Example

- Model: SM, Singlet extension, 2HDM, NMSSM
- Input: V_{tree}, num of fields, symmetry, field-dependent masses, Debye masses

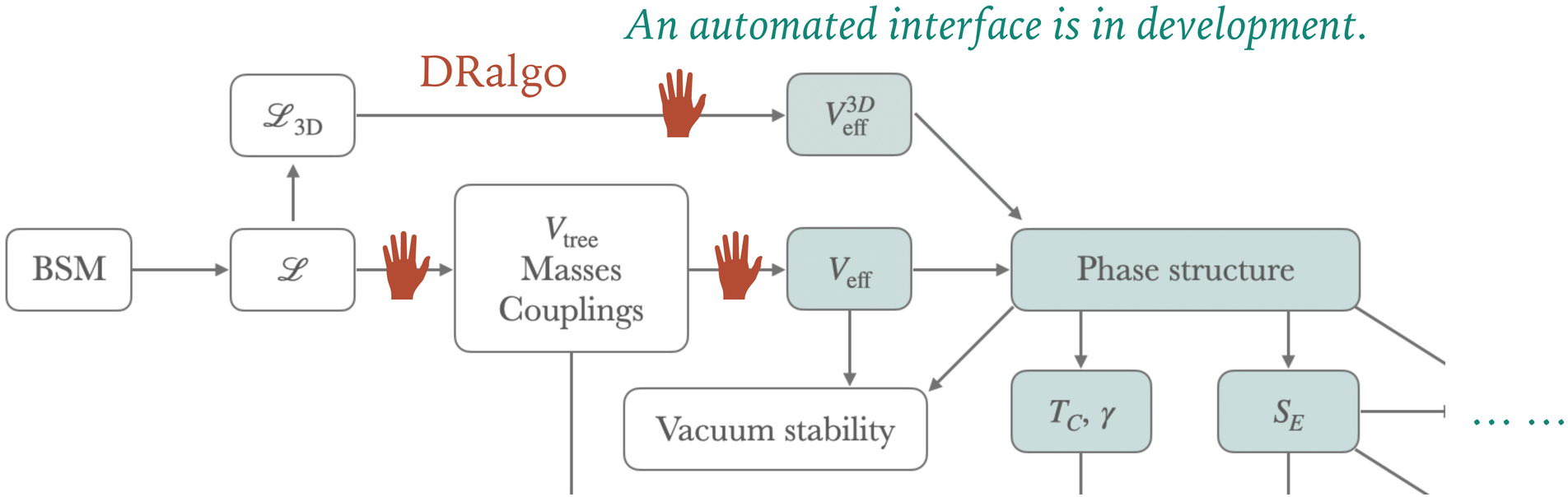
```
91   // W, Z, photon
92   std::vector<double> get_vector_debye_sq(Eigen::VectorXd phi, double T) const override {
93     const double h_sq = square(phi[0]);
94     const double T_sq = square(T);
95     const double MW_T_sq = 0.25 * square(SM_g) * h_sq;
96     const double MZ_T_sq = 0.25 * (square(SM_g) + square(SM_gp)) * h_sq;
97     const double Mphoton_T_sq = 0.;
98
99     const double MW_L_sq = 0.25 * square(SM_g) * h_sq + 11. / 6. * square(SM_g) * T_sq;
100    const double a_L = (square(SM_g) + square(SM_gp)) * (3. * h_sq + 22. * T_sq);
101    const double b_L = std::sqrt(9. * square(square(SM_g) + square(SM_gp)) * square(h_sq)
102                    + 132. * square(square(SM_g) - square(SM_gp)) * h_sq * T_sq
103                    + 484. * square(square(SM_g) - square(SM_gp)) * pow_4(T));
104    const double MZ_L_sq = (a_L + b_L) / 24.;
105    const double Mphoton_L_sq = (a_L - b_L) / 24.;
106
107    // Mphoton_sq must be put at the end, as it will not be used in the OSlike scheme.
108    return {MW_L_sq, MZ_L_sq, Mphoton_L_sq, MW_T_sq, MZ_T_sq, Mphoton_T_sq};
109  }
110
111  // W, Z, photon
112  std::vector<double> get_vector_dofs() const override {
113    return {2., 1., 1., 4., 2, 2};
114  }
```

$$
\begin{aligned}
m^2_{W_T}(\boldsymbol{\phi}) &= \frac{g^2}{4}\phi_h^2, \\
m^2_{W_L}(\boldsymbol{\phi}, T) &= \frac{g^2}{4}\phi_h^2 + \frac{11}{6}g^2T^2, \\
m^2_{\gamma_T}(\boldsymbol{\phi}) &= 0, \\
m^2_{\gamma_L}(\boldsymbol{\phi}, T) &= \frac{1}{24}\Big[\left(g^2+g'^2\right)\left(3\phi_h^2+22T^2\right) \\
&\qquad - \sqrt{9\left(g^2+g'^2\right)^2\phi_h^4 + 44T^2\left(g^2-g'^2\right)^2\left(3\phi_h^2+11T^2\right)}\Big], \\
m^2_{Z_T}(\boldsymbol{\phi}) &= \frac{g^2+g'^2}{4}\phi_h^2, \\
m^2_{Z_L}(\boldsymbol{\phi}, T) &= \frac{1}{24}\Big[\left(g^2+g'^2\right)\left(3\phi_h^2+22T^2\right) \\
&\qquad + \sqrt{9\left(g^2+g'^2\right)^2\phi_h^4 + 44T^2\left(g^2-g'^2\right)^2\left(3\phi_h^2+11T^2\right)}\Big].
\end{aligned}
$$

$$
n_{W_L} = 2,\ n_{Z_L} = 1,\ n_{\gamma_L} = 1, n_{W_T} = 4,\ n_{Z_T} = 2,\ n_{\gamma_T} = 2
$$

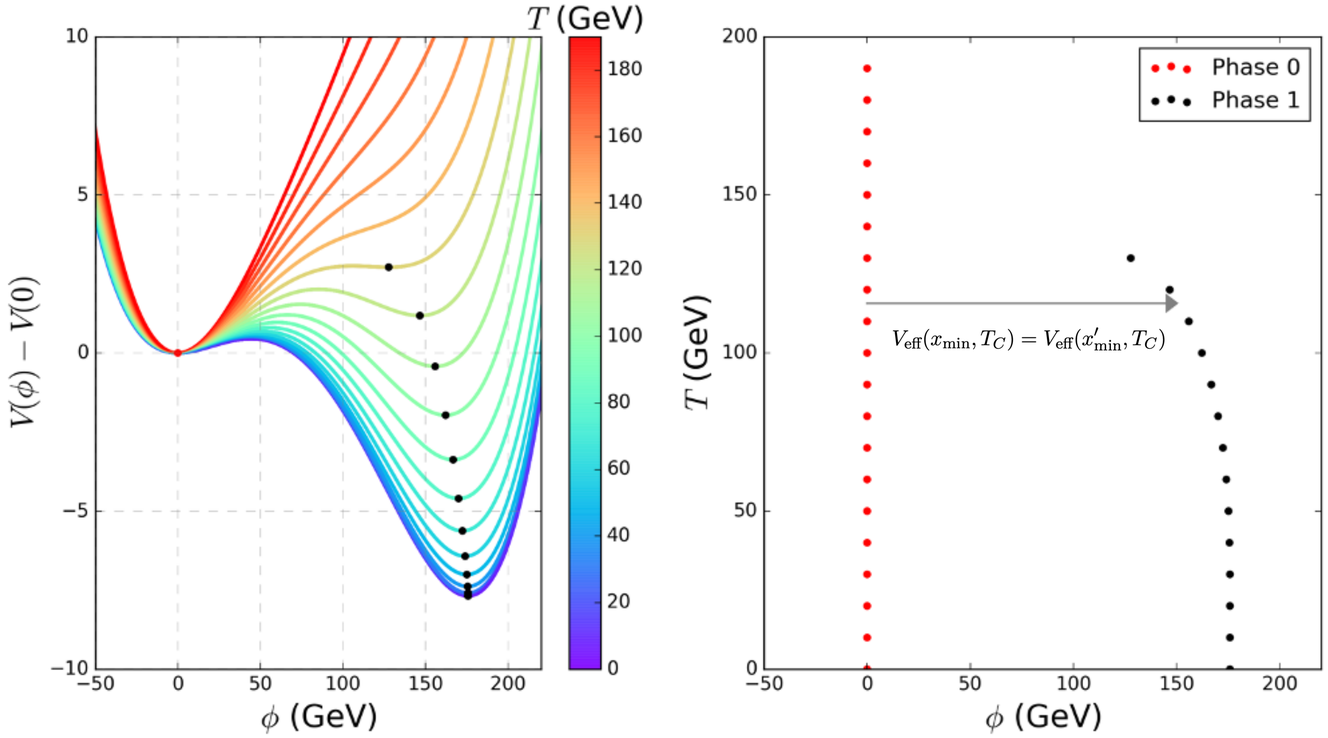
Example

- Model: SM, Singlet extension, 2HDM, NMSSM
- Input: V_{eff}^{3D} from DRalgo



Outputs of PhaseTracer2

- Phase structure



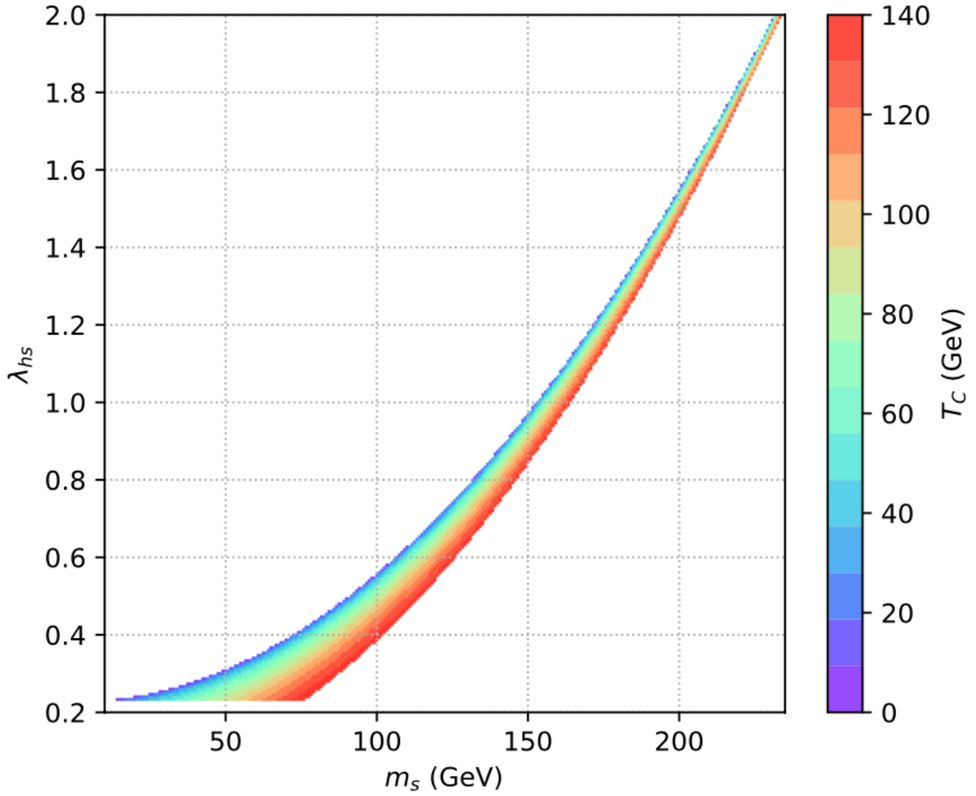
PhaseTracer — -zsh — 58×43

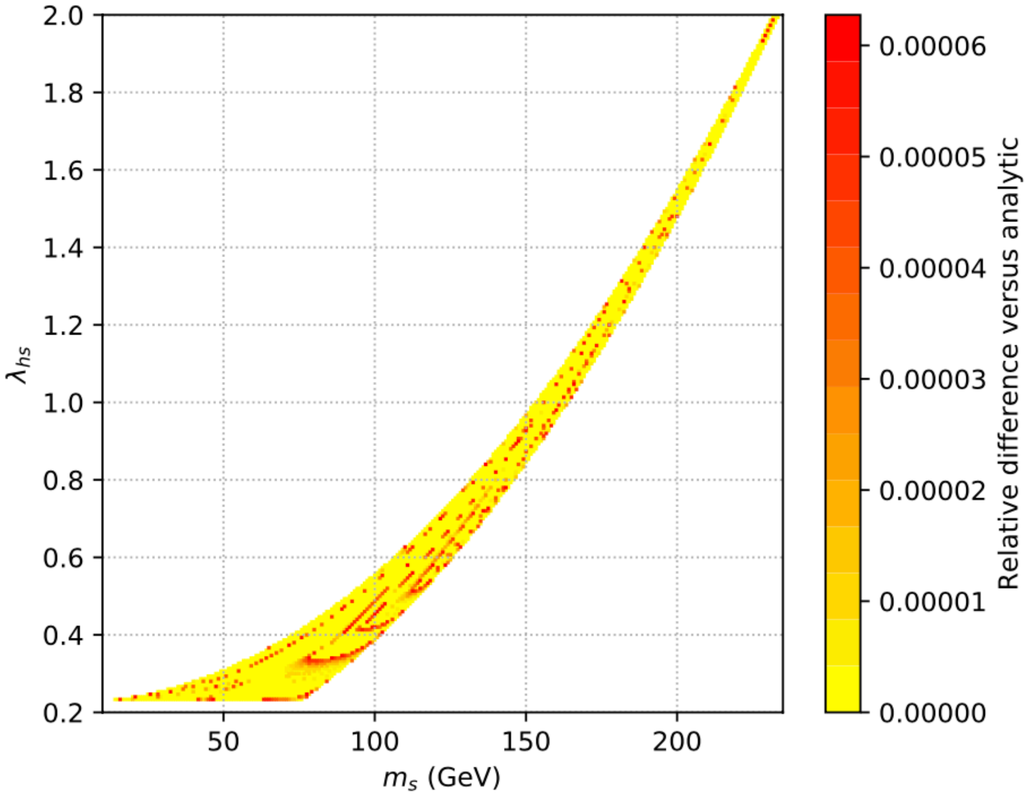
```
zy@192 PhaseTracer % ./bin/run_1D_test_model
found 2 phases

=== phase key = 0 ===
Maximum temperature = 1000
Minimum temperature = 33.1513
Field at tmax = [-7.71149e-06]
Field at tmin = [-4.80989e-06]
Potential at tmax = 5.94077e-06
Potential at tmin = 2.29061e-10
Ended at tmax = Reached tstop
Ended at tmin = Jump in fields indicated end of phase

=== phase key = 1 ===
Maximum temperature = 61.7437
Minimum temperature = 0
Field at tmax = [37.8319]
Field at tmin = [81.1597]
Potential at tmax = 65886.6
Potential at tmin = -1.66587e+06
Ended at tmax = Jump in fields indicated end of phase
Ended at tmin = Reached tstop

found 1 transition

=== transition from phase 0 to phase 1 ===
changed = [true]
TC = 59.1608
false vacuum (TC) = [5.69488e-06]
true vacuum (TC) = [50.0002]
gamma (TC) = 0.845158
delta potential (TC) = 0.00117802
TN = 57.3828
false vacuum (TN) = [6.40552e-06]
true vacuum (TN) = [53.6195]

=== Gravitational wave generated at T = 57.3828
    alpha = 0.00138966
    beta over H = 7118.31
    peak_frequency = 0.121947
    peak_amplitude = 3.59588e-23
    siganl to noise ratio for LISA = 3.72509e-13
zy@192 PhaseTracer %
```

Outputs of PhaseTracer

- Critical temperature T_C

$$V_{\text{eff}}(x_{\text{min}}, T_C) = V_{\text{eff}}(x'_{\text{min}}, T_C)$$





Outputs of PhaseTracer2

- Bounce action S_E

$$\frac{\Gamma}{V} = A(T)e^{-S_E/T}[1 + \mathcal{O}(\hbar)]$$

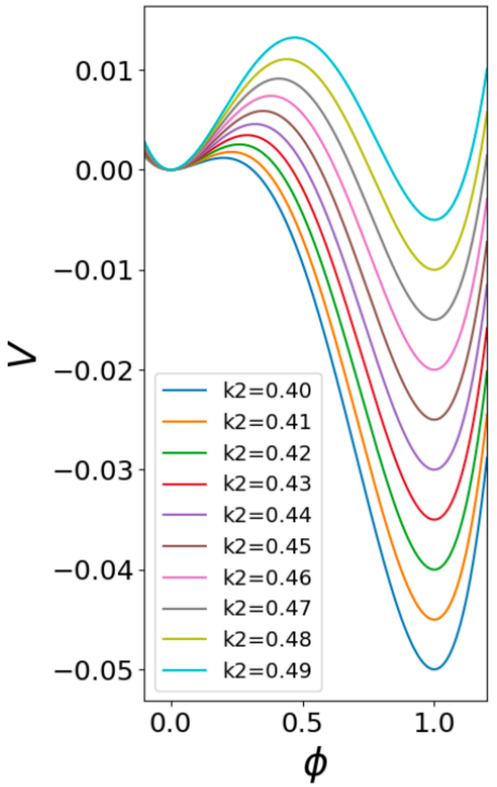
$$S_E = \mathcal{S}_{d-1}\int_0^\infty \rho^{d-1}\left(\frac{1}{2}\dot{\phi}^2 + V(\phi)\right)$$

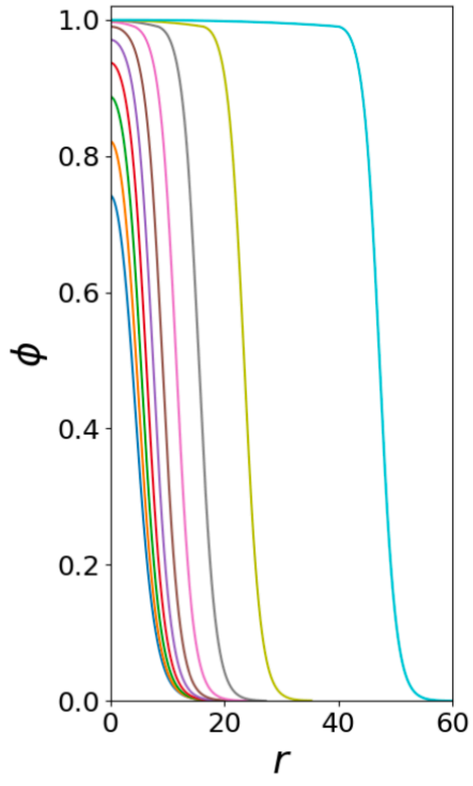
$$\ddot{\phi}(\rho) + \frac{d-1}{\rho}\dot{\phi}(\rho) = V'(\phi)$$

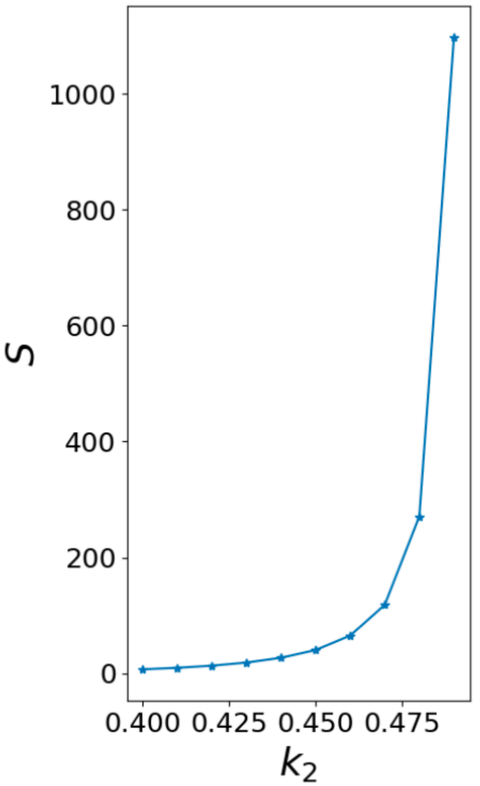
$$\dot{\phi}(\rho) = 0$$

$$\phi(\rho \to \infty) = \phi_f$$

$$\dot{\phi}(\rho \to \infty) = 0$$





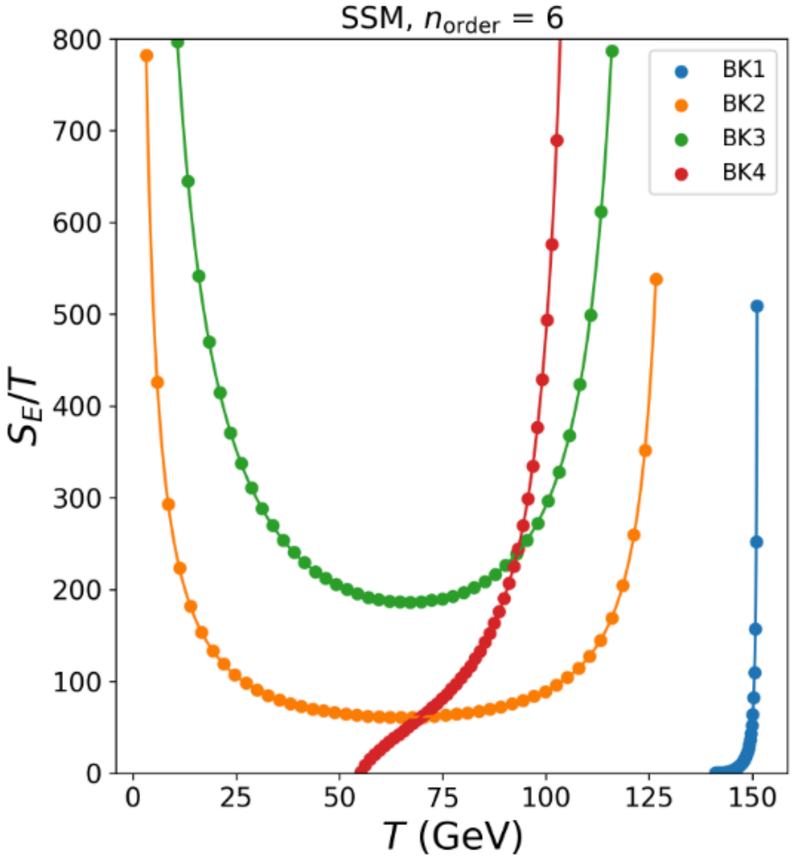


Path Deformation

BubbleProfiler

Outputs of PhaseTracer2

- Nucleation temperature T_N, strength factor α, transition's duration



$$N(T) = \int_{T_C}^{T} dT' \frac{\Gamma(T')P_f(T')}{T'H^4(T')}$$

$$\frac{S_E(T_N)}{T_N} \sim 140$$

$$\alpha = \frac{D\theta}{\pi^2 g_* T_*^4/30} = \frac{1}{\pi^2 g_* T_*^4/30}\left(V(\phi) - \frac{T}{4}\frac{\partial V(\phi,T)}{\partial T}\right)\Bigg|_{\phi_t}^{\phi_f}$$

$$\beta(T) = \frac{d}{dt}\left(\frac{S_E}{T}\right) = TH(T)\frac{d}{dT}\left(\frac{S_E}{T}\right)$$

+ TransitionSlover

Outputs of PhaseTracer2

➤ Gravitational Wave

- Bubble collisions

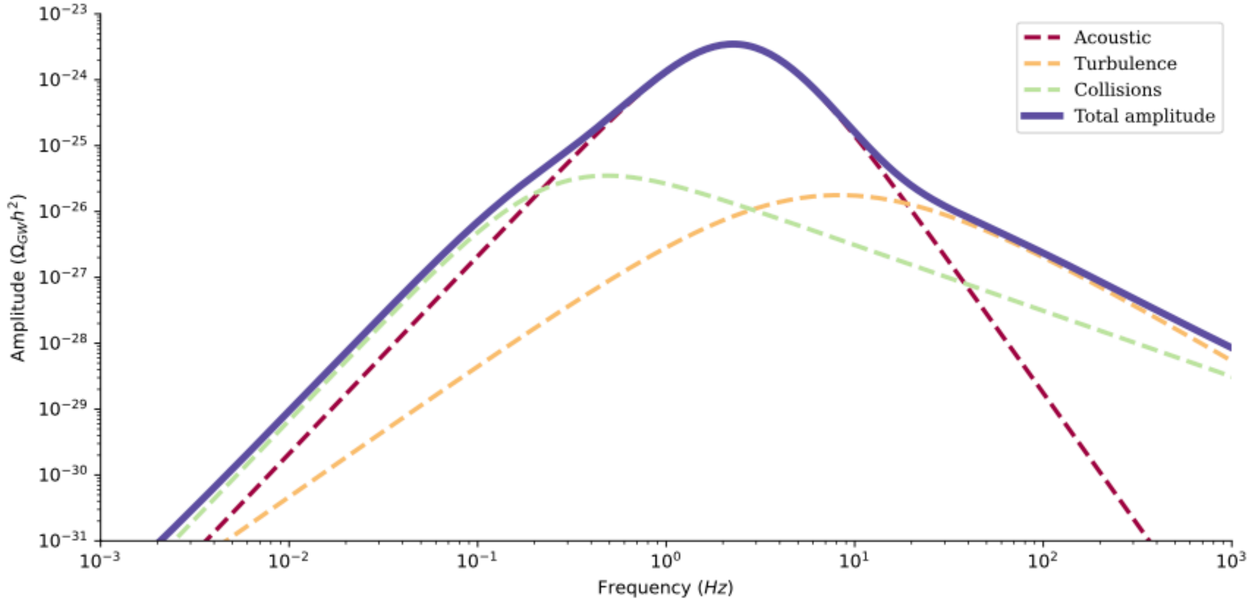
$$\Omega_{\mathrm{col}}^{\mathrm{env}} h^2 = 1.67 \times 10^{-5} \Delta \left(\frac{100}{g_*}\right)^{\frac{1}{3}} \left(\frac{H_*}{\beta}\right)^2 \left(\frac{\kappa_\phi \alpha}{1+\alpha}\right)^2 \times S_{\mathrm{env}}(f), \tag{82}$$

- Turbulence

$$\Omega_{\mathrm{turb}} h^2 = 3.35 \times 10^{-4} \left(\frac{H_*}{\beta}\right) \left(\frac{\kappa_{\mathrm{turb}} \alpha}{1+\alpha}\right)^{3/2} \left(\frac{100}{g_*}\right)^{1/3} v_w \times \frac{(f/f_{\mathrm{turb}})^3}{[1+(f/f_{\mathrm{turb}})]^{11/3} (1+8\pi f/H_0)}, \tag{88}$$

- Sound waves

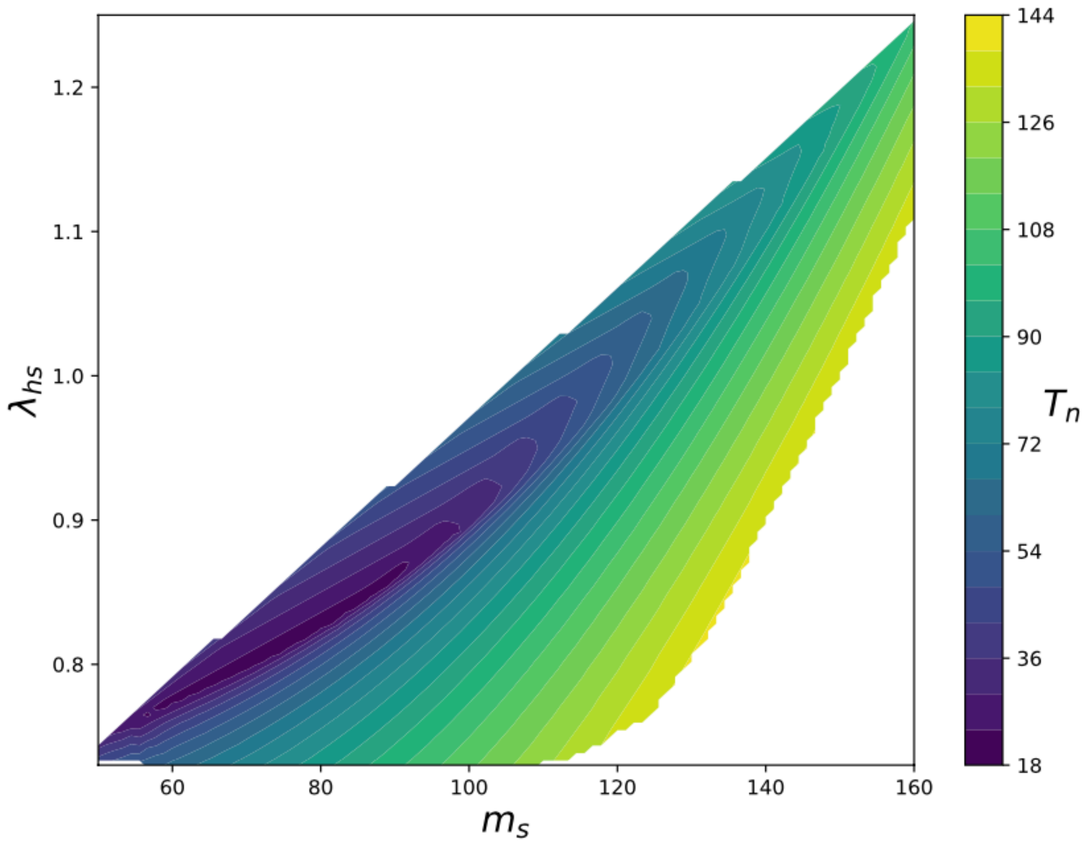
$$\Omega_{\mathrm{sw}} h^2 = 2.061\, F_{\mathrm{gw},0}\, \Gamma^2\, \bar{U}_f^4\, S_{\mathrm{sw}}(f)\, \tilde{\Omega}_{\mathrm{gw}} \times \min(H_* R_* / \bar{U}_f, 1)\ (H_* R_*)\, h^2, \tag{91}$$

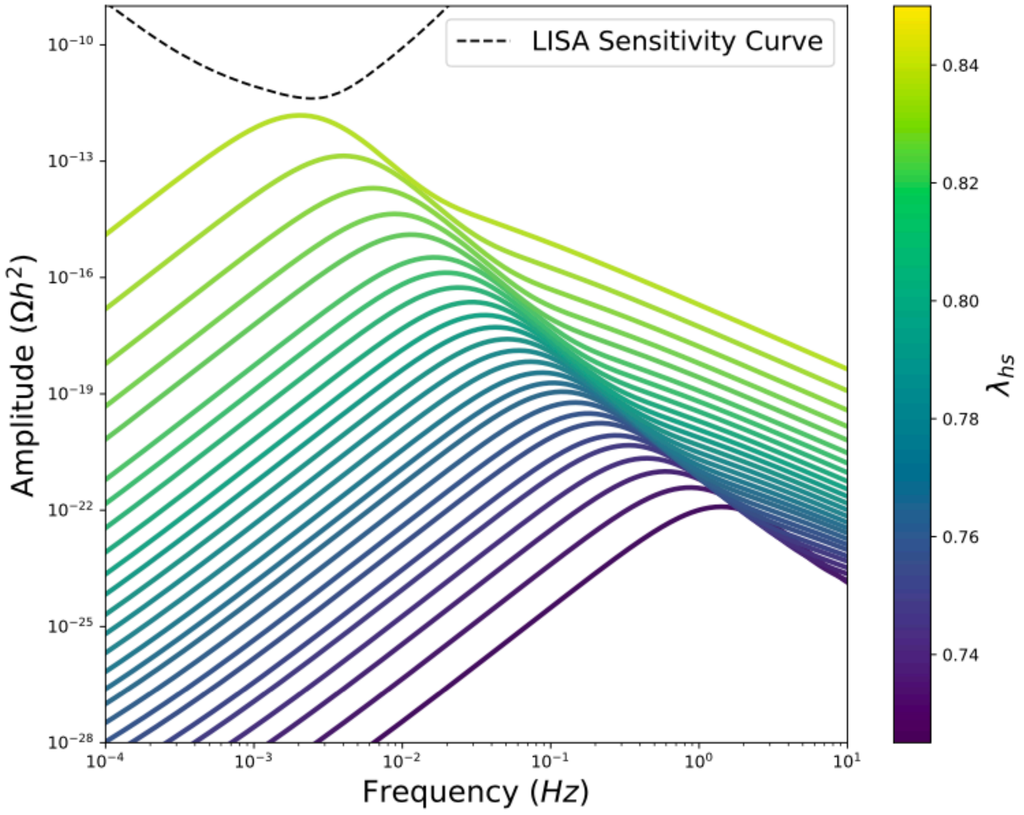


+ TransitionSlover

Outputs of PhaseTracer2

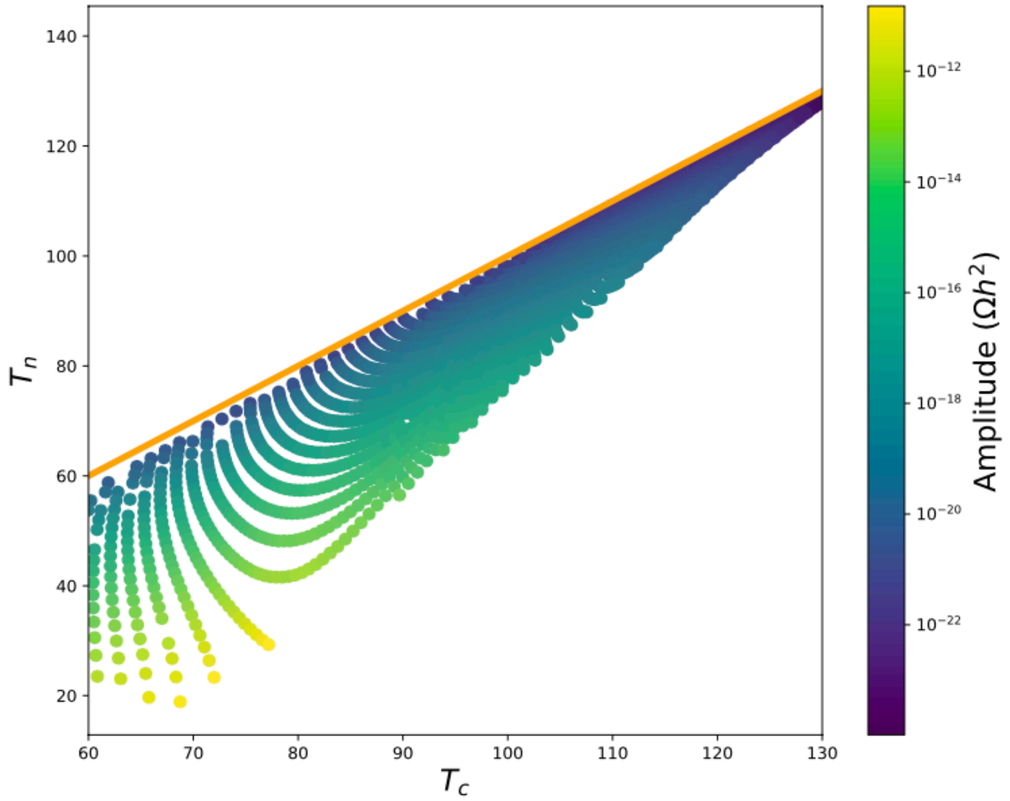
- Gravitational Wave

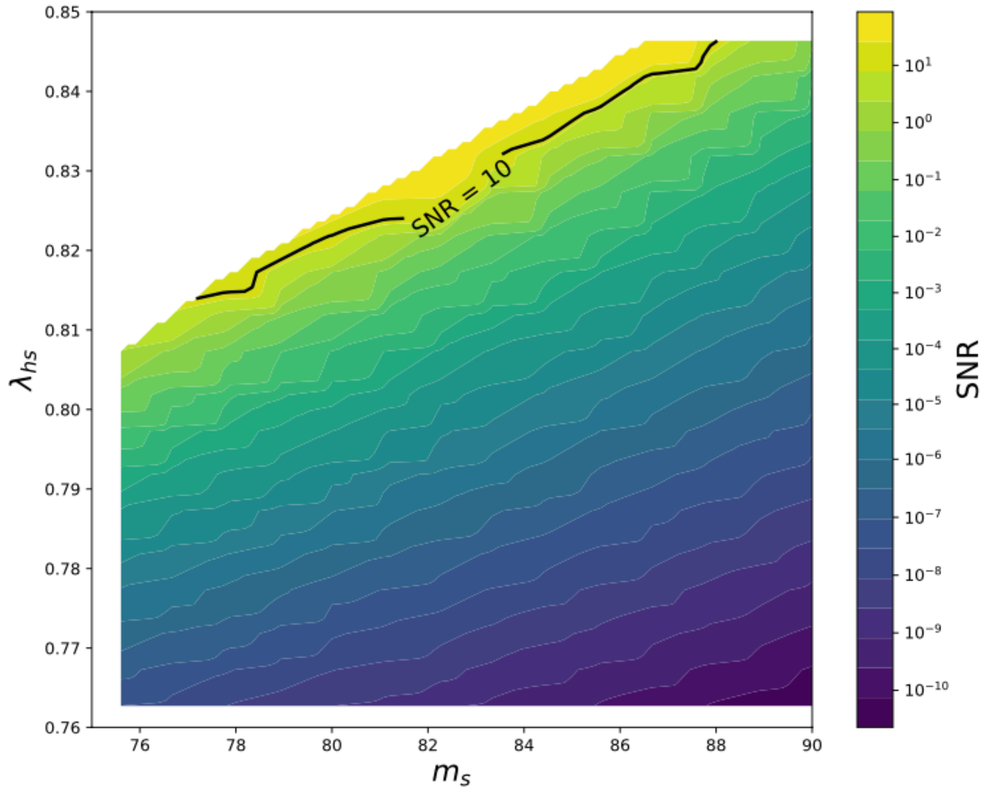




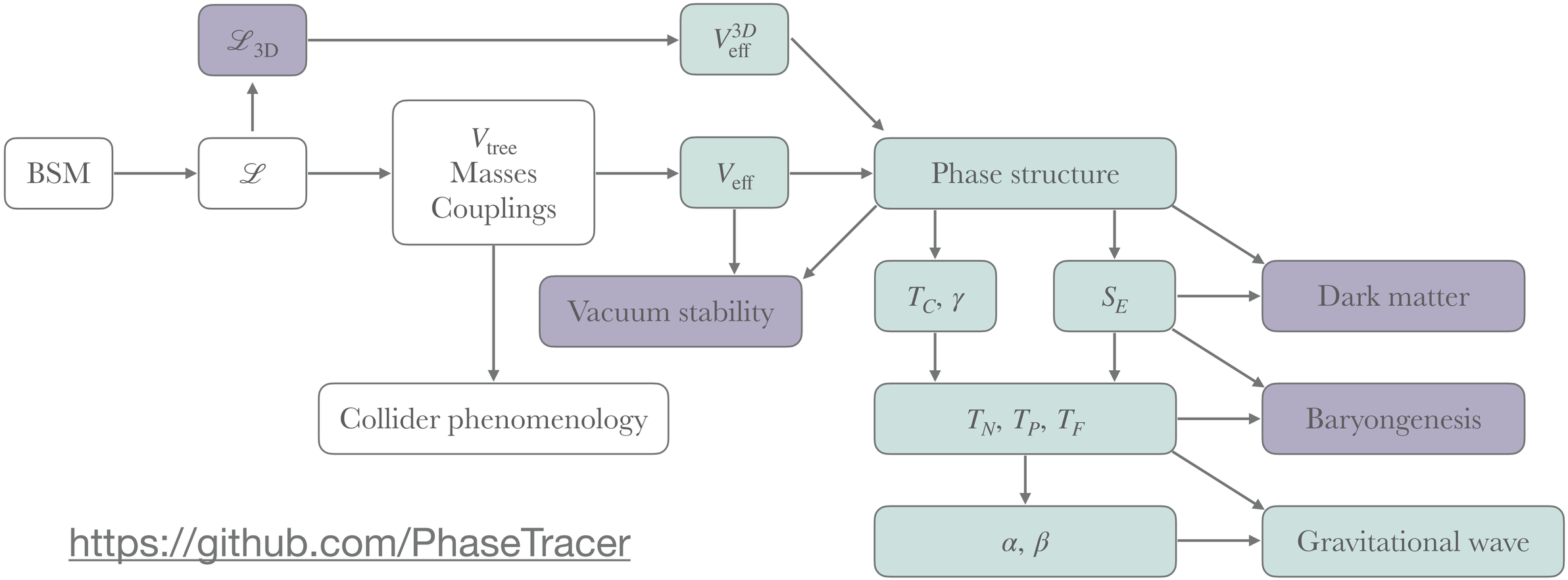
Outputs of PhaseTracer2

- Gravitational Wave





Summary and outlook



https://github.com/PhaseTracer

Thank you!

PhaseTracer2 vs. CosmoTransitions

- Nucleation temperature T_N

	T_C	T_N	False VEV	True VEV	Time
`CosmoTransitions`	109.4	84.24	$[231.1, -136.8]$	$[286.4, 382.2]$	7.62 s
`PhaseTracer`	109.4	84.25	$[231.1, -136.8]$	$[286.4, 382.2]$	4.70 s