# Allgemeine Relativitätstheorie mit dem Computer

*PC-POOL RAUM 01.120*
*JOHANN WOLFGANG GOETHE UNIVERSITÄT*
*20. JUNI, 2017*

*MATTHIAS HANAUSKE*

*FRANKFURT INSTITUTE FOR ADVANCED STUDIES*
*JOHANN WOLFGANG GOETHE UNIVERSITÄT*
*INSTITUT FÜR THEORETISCHE PHYSIK*
*ARBEITSGRUPPE RELATIVISTISCHE ASTROPHYSIK*
*D-60438 FRANKFURT AM MAIN*
*GERMANY*

## 9. Vorlesung

# Allgemeines zur Vorlesung, Plan für die heutige Vorlesung

- Kompensationstermine der Vorlesungen 26.05.2017 und 14.07.2017: **Dienstag der 20.06. und 27.06.2017, jeweils um 12.15-13.45 Uhr.**
- Kompensationstermine der Vorlesung am 30.06.2017: **Dienstag der 04.07.2017 um 12.15-13.45 Uhr.**
- Wiederholung: Das parallele C++ Programm zum Berechnen der Tolman-Oppenheimer-Volkoff (TOV) Gleichungen einer Sequenz von Neutronen/Quark Sternen.
- Die OpenMP- und MPI- C++ Versionen mit geordneter Ausgabe in eine Datei, variabler Zustandsgleichung und Terminalausgabe der benötigten Zeit.
- Einführung in Teil III: Das Einstein Toolkit

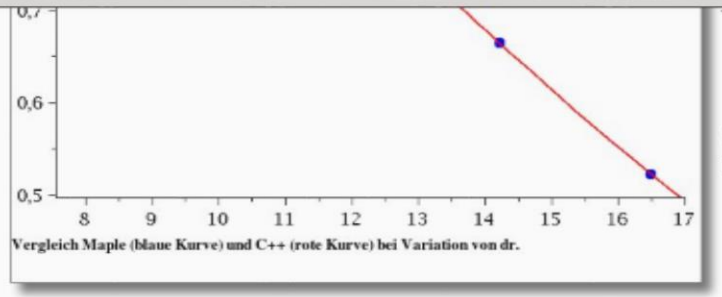
# Wiederholung: Einführung in die Parallele Programmierung

fias.uni-frankfurt.de/~hanauske/VARTC/T2/intro/Hanauske_ParallelizationTut.odp
fias.uni-frankfurt.de/~hanauske/VARTC/T2/intro/Hanauske_ParallelizationTut.pdf

## Introduction

1. Parallelization on shared memory systems using OpenMP
2. Parallelization on distributed memory systems using MPI
3. Further resources

# Parallele Programme siehe Teil 2 der Internetseite der Vorlesung



Vergleich Maple (blaue Kurve) und C++ (rote Kurve) bei Variation von dr.

**2.3) Parallele OpenMP-Version 1 von 2.2)**

Das sequentielle Programm 2.2) wurde nun mittels OpenMP (siehe TOV OpenMP Version) parallelisiert. Hiebei wurde einfach das OpenMP-Pragma *'#pragma omp parallel for private(i,M,p,e,r,nu,dM,dp,de,dnu)'* vor die for-Schleife der unabhängigen Berechnung der einzelnen Neutronensterne geschrieben. Wichtig ist nun, dass man das Programm mit dem folgenden Befehl compiliert: 'c++ -fopenmp TOV_parallel_omp.cpp'. Führt man das Programm mit './a.out' aus, so erkennt man als erstes, dass es (in Abhängigkeit wieviele CPU-Kerne man in seinem Computer hat), viel schneller läuft. Die im Terminal ausgegebenen Werte sind jedoch nicht mehr geordnet, sondern die Berechnung der 40 Neutronensterne erfolgt parallel und ungeordnet. Die nebenstehende Abbildung zeigt die Terminalausgabe des parallelen Programms und die Auslastung der 8 CPU-Kerne meines Laptops, wobei zuerst das parallele Programm und dannach das sequentielle ausgeführt wurde.

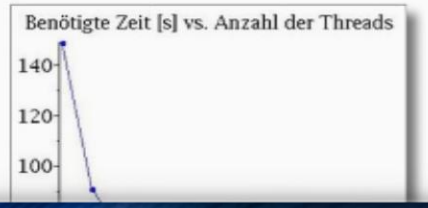
```
Datei  Bearbeiten  Ansicht  Lesezeichen  Einstellungen  Hilfe
hanauske@ITPRelAstro-Aspire-VN7-591G:~$ c++ -fopenmp TOV_parallel_omp.cpp
hanauske@ITPRelAstro-Aspire-VN7-591G:~$ ./a.out
35
Neutronensternradius [km]           = 9.13464
Neutronensternmasse [Sonnenmassen]  = 1.20785
00-Metrikkomponente im Sternzentrum = 0.198145
30
Neutronensternradius [km]           = 9.50537
Neutronensternmasse [Sonnenmassen]  = 1.21506
00-Metrikkomponente im Sternzentrum = 0.220643
25
Neutronensternradius [km]           = 9.94998
Neutronensternmasse [Sonnenmassen]  = 1.21927
00-Metrikkomponente im Sternzentrum = 0.248056
20
Neutronensternradius [km]           = 10.4976
Neutronensternmasse [Sonnenmassen]  = 1.21836
00-Metrikkomponente im Sternzentrum = 0.28223
15
Neutronensternradius [km]           = 11.1977
Neutronensternmasse [Sonnenmassen]  = 1.20841
00-Metrikkomponente im Sternzentrum = 0.326145
10
Neutronensternradius [km]           = 12.1444
```


**2.4) Parallele OpenMP-Version 2 ( 2.3) mit geordneter Terminal-Ausgabe )**

Diese Version entspricht Version 2.3) mit einer geordneten Ausgabe. Die geordnete Ausgabe wird hierbei realisiert, indem für die drei ausgegebenen, numerischen Werte (Radius, Masse, zentraler $g_{00}$-Wert), drei Datenfelder (Arrays) der Länge 40 eingerichtet werden. Nachdem ein OpenMP-Thread mit seiner Berechnung fertig ist, speichert er sein individuelles Ergebnis in die spezifische Position innerhalb des Arrays und berechnet den nächsten Stern. Die geordnete Ausgabe aller Werte erfolgt dann sequentiell, außerhalb der parallelisierten Schleife.

**2.5) Parallele OpenMP-Version 3 ( 2.4) mit Ausgabe in eine Datei )**

Diese Version entspricht Version 2.4) wobei die geordnete Ausgabe nun nicht mehr in dem Terminal geschieht, sondern die berechneten Werte werden in eine externe Datei ( tov.txt ) ausgegeben - die Ausgabedatei erfolgt im Unterordner 'output', welcher vor dem Ausführen des Programms angelegt werden muss. Der Vorteil hierbei ist, dass nachdem das Programm ausgefürt wurde, die Ergebnisse einfacher verarbeitet und dargestellt werden können. So kann man z.B. mittels Gnuplot sich das Radius-Masse Diagramm darstellen. Noch einfacher, kann man sich die einzelnen Gnuplot-Befehle zum Darstellen diverser Diagramme in ein ausführbares Shell-Script schreiben, das dann automatisch die jeweiligen Plots erzeugt (siehe Gnuplot Shell-Script).



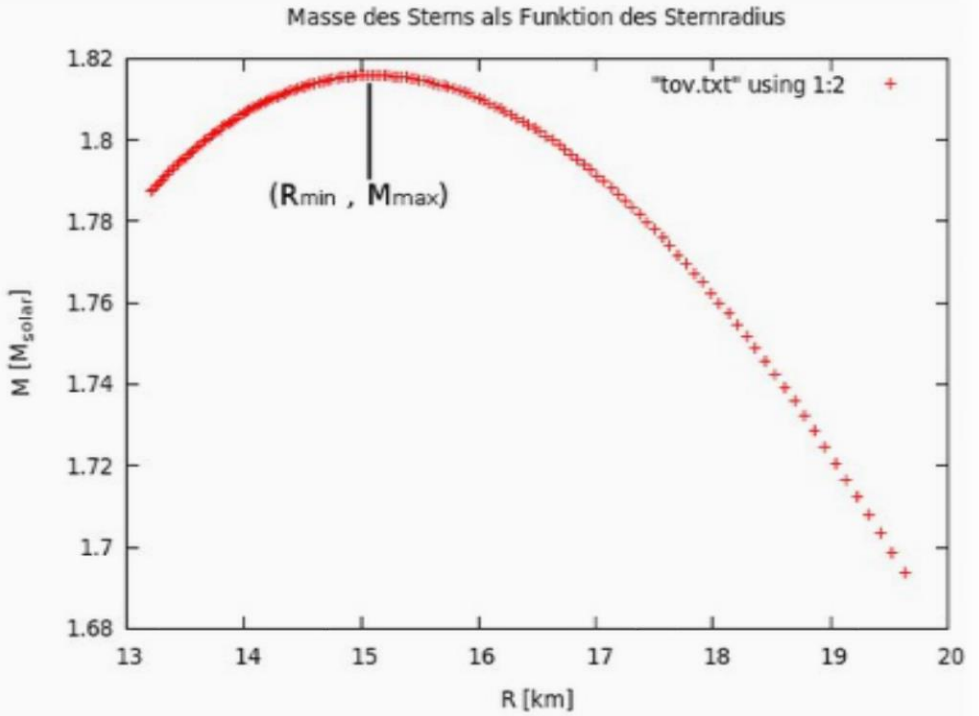

**2.6) Parallele OpenMP-Version mit geordneter Ausgabe in eine Datei und variabler Zustandsgleichung**

Diese Version entspricht Version 2.5), wobei die als Funktion definierte Zustandsgleichung variabler gestaltet wurde (EOS: ( $e(P, K, \gamma) = (P/K)^{1/\gamma}$ ) für Neutronensterne und Weiße Zwerge bzw. ( e(P,Bag)=3 p + 4 Bag ) für Quarksterne im MIT-Bag Model).

**Struktur und Performance des parallelen OpenMP - C++ Programms**

# Aufgaben zum 2.Teil der Vorlesung siehe E-Learning „Lon Capa“

Berechnen Sie unter Verwendung des C++ Programms aus Teil II der Vorlesung die maximale Masse $M_{max}$ in $[\mathrm{M}_\odot]$ und den zugehörigen minimalen Radius $R_{min}$ eines Neutronensterns in [km]. Verwenden Sie eine polytrope Zustandsgleichung der Form $p = K * e^{\gamma}$, wobei $\gamma = 5/3$ und $K = 20.25 \left[\mathrm{km}^{4/3}\right]$ ist.

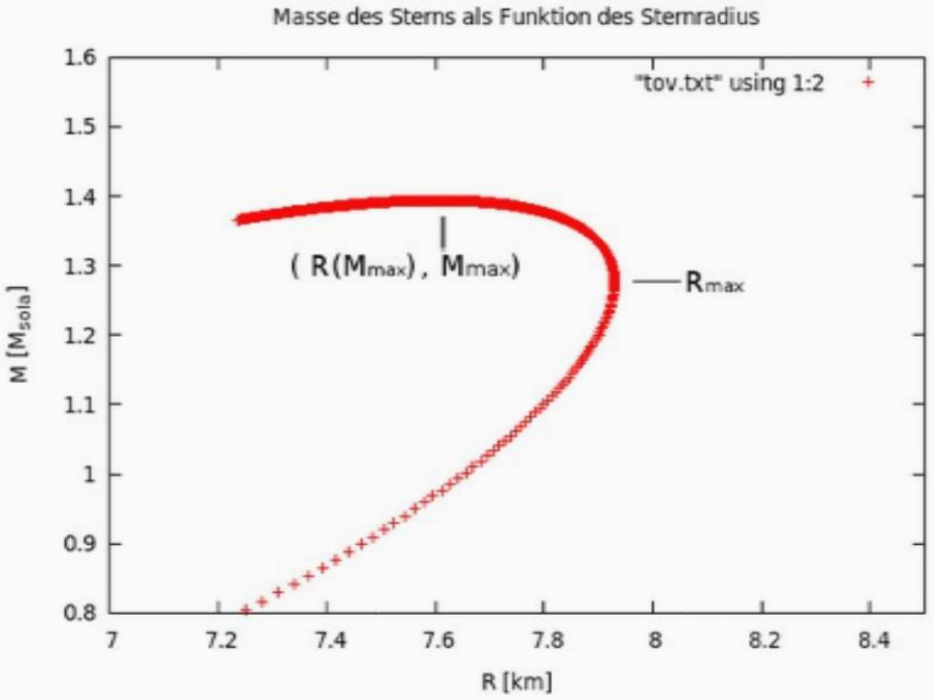



$M_{max}$ = _____ , $R_{min}$ = _____

Antwort einreichen  Versuche 0/20

# Aufgaben zum 2.Teil der Vorlesung siehe E-Learning „Lon Capa“

Berechnen Sie unter Verwendung des C++ Programms aus Teil II der Vorlesung die maximale Masse $M_{max}$ in $[\mathrm{M}_\odot]$ und den zugehörigen Radius $R(M_{max})$ eines Quarkstern Modells in [km]. Verwenden Sie die lineare Zustandsgleichung des MIT-Bag Modells $p = \frac{1}{3}\left(e - 4 * B\right)$;, wobei der Parameter $B$ die für das Confinement nötige Bag Konstante ist; verwenden Sie $B$=0.000152869496944 (entspricht ungefähr $B^{1/4} =$ 194 [MeV]). Geben Sie desweiteren auch dem maximalen Radius $R_{max}$ des Quarksternmodells an.

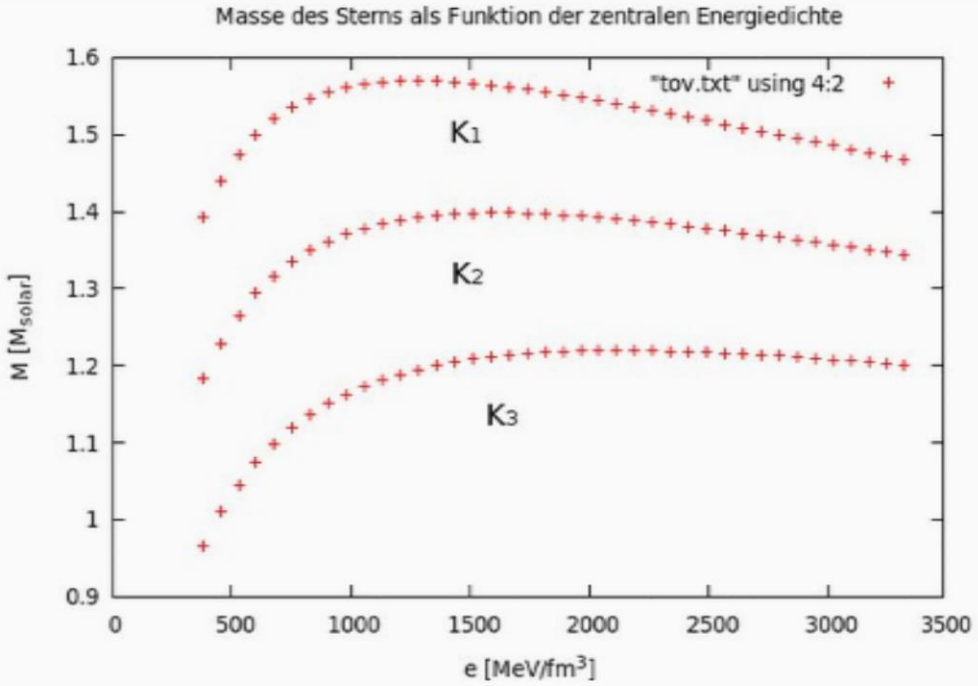



$M_{max}$ = , $R(M_{max})$ = , $R_{max}$ =

Antwort einreichen Versuche 0/20

# Aufgaben zum 2.Teil der Vorlesung siehe E-Learning „Lon Capa“

Die maximale Masse $M_{max}$ eines Neutronensterns sei gegeben, und die zugrunde liegende Zustandsgleichung der Neutronensternmaterie sei durch folgenden polytropen Ansatz $p = K * e^{\gamma}$ bestimmt, wobei $\gamma = 5/3$ und $K =$ eine noch zu bestimmende unbekannte Konstante ist. Bei Variation von $K$ ändert sich das gesamte Masse-Radius, bzw. Masse-zentrale Energiedichte Profil einer Sequenz von Sternen und der Wert der maximale Masse $M_{max}$ verschiebt sich (siehe nebenstehende Abbildung). Berechnen Sie unter Verwendung des C++ Programms aus Teil II der Vorlesung den Wert der Konstanten $K$ in $\left[\mathrm{km}^{4/3}\right]$ und geben Sie den zugehörigen Radius des maximalen Massen Sterns ($R_{M_{max}}$) an. Der Wert der maximalen Masse beträgt $M_{max} = 1.735147\ [\mathrm{M}_{\odot}]$.




$K =$ , $R_{M_{max}} =$

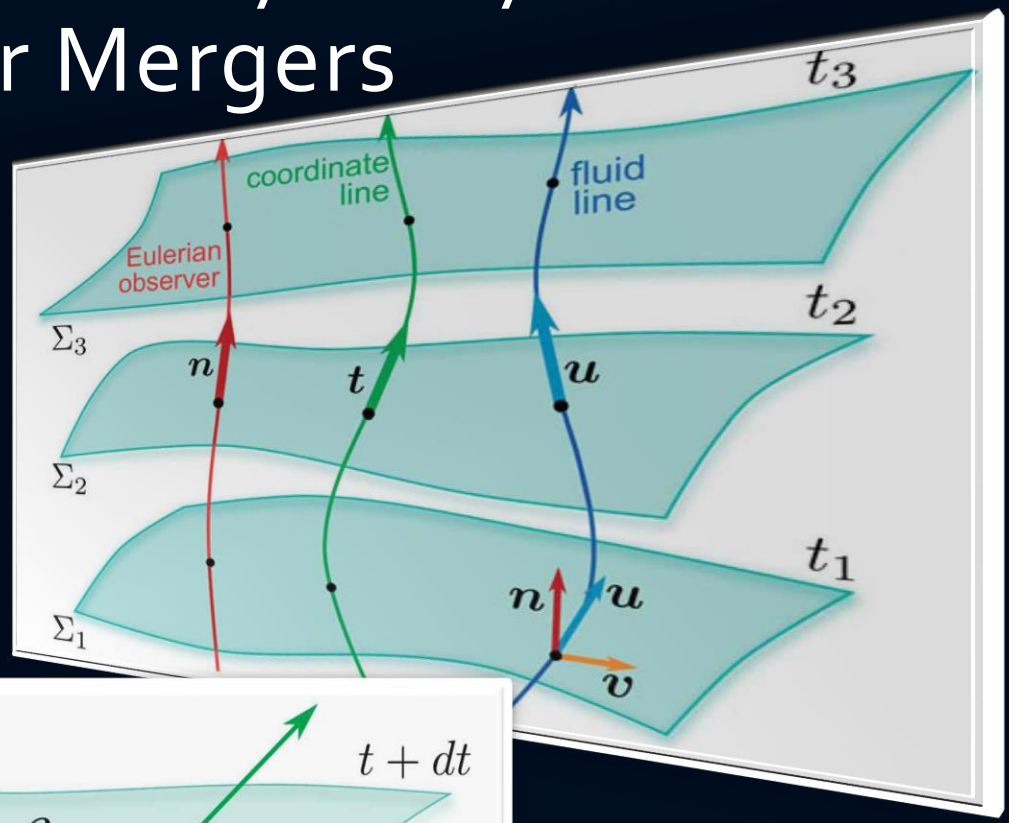
Antwort einreichen  Versuche 0/20

# Numerical Relativity and Relativistic Hydrodynamics of Binary Neutron Star Mergers

A realistic numerical simulation of a twin star collapse, a merger of two compact stars or a collapse to a black hole needs to go beyond a static, spherically symmetric TOV-solution of the Einstein- and hydrodynamical equations.

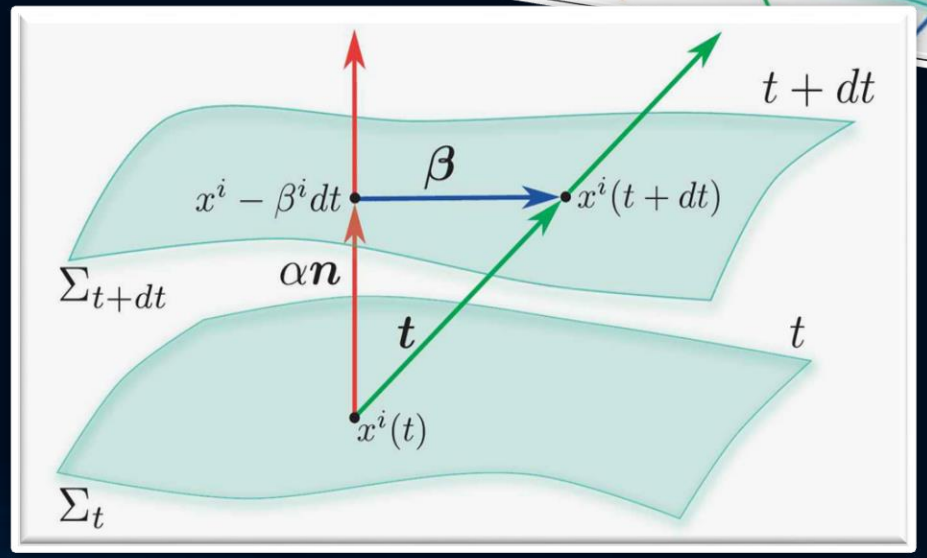
$$R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R = 8\pi T_{\mu\nu}$$

$$\nabla_\mu (\rho u^\mu) = 0\,, \qquad \nabla_\nu T^{\mu\nu} = 0\,.$$




(3+1) decomposition of spacetime

$$g_{\mu\nu} = \begin{pmatrix} -\alpha^2 + \beta_i \beta^i & \beta_i \\ \beta_i & \gamma_{ij} \end{pmatrix}$$




$$d\tau^2 = \alpha^2(t, x^j) dt^2$$

$$x^i_{t+dt} = x^i_t - \beta^i(t, x^j) dt$$

All figures and equations from: Luciano Rezzolla, Olindo Zanotti: Relativistic Hydrodynamics, Oxford Univ. Press, Oxford (2013)

# The ADM equations

The ADM (Arnowitt, Deser, Misner) equations come from a reformulation of the Einstein equation using the (3+1) decomposition of spacetime.

$$\partial_t \gamma_{ij} = -2\alpha K_{ij} + \mathscr{L}_{\boldsymbol{\beta}} \gamma_{ij} = -2\alpha K_{ij} + D_i \beta_j + D_j \beta_i$$

$$\partial_t K_{ij} = -D_i D_j \alpha + \beta^k \partial_k K_{ij} + K_{ik} \partial_j \beta^k + K_{kj} \partial_i \beta^k + \alpha \left( {}^{(3)}R_{ij} + K K_{ij} - 2 K_{ik} K^k{}_j \right) + 4\pi\alpha \left[ \gamma_{ij} (S - E) - 2 S_{ij} \right]$$

Time evolving part of ADM

$$D_j (K^{ij} - \gamma^{ij} K) = 8\pi S^i$$

$${}^{(3)}R + K^2 - K_{ij} K^{ij} = 16\pi E$$

Constraints on each hypersurface

Three dimensional covariant derivative

$$D_\nu := \gamma^\mu{}_\nu \nabla_\mu = (\delta^\mu_\nu + n_\nu n^\mu) \nabla_\mu$$

Three dimensional Riemann tensor

$${}^{(3)}R^\mu{}_{\nu\kappa\sigma} = \partial_\kappa {}^{(3)}\Gamma^\mu_{\nu\sigma} - \partial_\sigma {}^{(3)}\Gamma^\mu_{\nu\kappa} + {}^{(3)}\Gamma^\mu_{\lambda\kappa} {}^{(3)}\Gamma^\lambda_{\nu\sigma} - {}^{(3)}\Gamma^\mu_{\lambda\sigma} {}^{(3)}\Gamma^\lambda_{\nu\kappa}$$

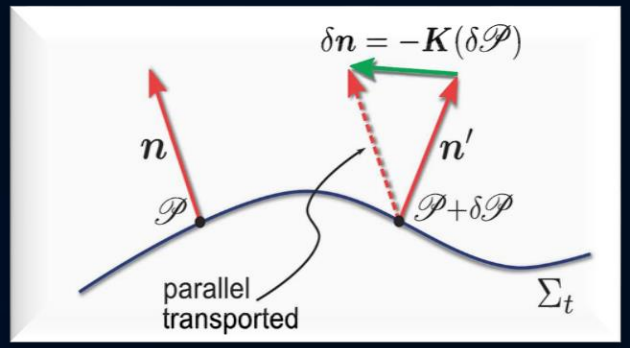
$${}^{(3)}\Gamma^\alpha_{\beta\gamma} = \frac{1}{2} \gamma^{\alpha\delta} \left( \partial_\beta \gamma_{\gamma\delta} + \partial_\gamma \gamma_{\delta\beta} - \partial_\delta \gamma_{\beta\gamma} \right)$$

Spatial and normal projections of the energy-momentum tensor:

$$S_{\mu\nu} := \gamma^\alpha{}_\mu \gamma^\beta{}_\nu T_{\alpha\beta}\,,$$
$$S_\mu := -\gamma^\alpha{}_\mu n^\beta T_{\alpha\beta}\,,$$
$$S := S^\mu{}_\mu\,,$$
$$E := n^\alpha n^\beta T_{\alpha\beta}\,,$$

Extrinsic Curvature:

$$K_{\mu\nu} := -\gamma^\lambda{}_\mu \nabla_\lambda n_\nu$$




All figures and equations from: Luciano Rezzolla, Olindo Zanotti: Relativistic Hydrodynamics, Oxford Univ. Press, Oxford (2013)

# From ADM to BSSNOK

Unfortunately the ADM equations are only weakly hyperbolic (mixed derivatives in the three dimensional Ricci tensor) and therefore not "well posed". It can be shown that by using a conformal traceless transformation, the ADM equations can be written in a hyperbolic form. This reformulation of the ADM equations is known as the BSSNOK (Baumgarte, Shapiro, Shibata, Nakamuro, Oohara, Kojima) formulation of the Einstein equation. Most of the numerical codes use this (or even better the CCZ4) formulation.

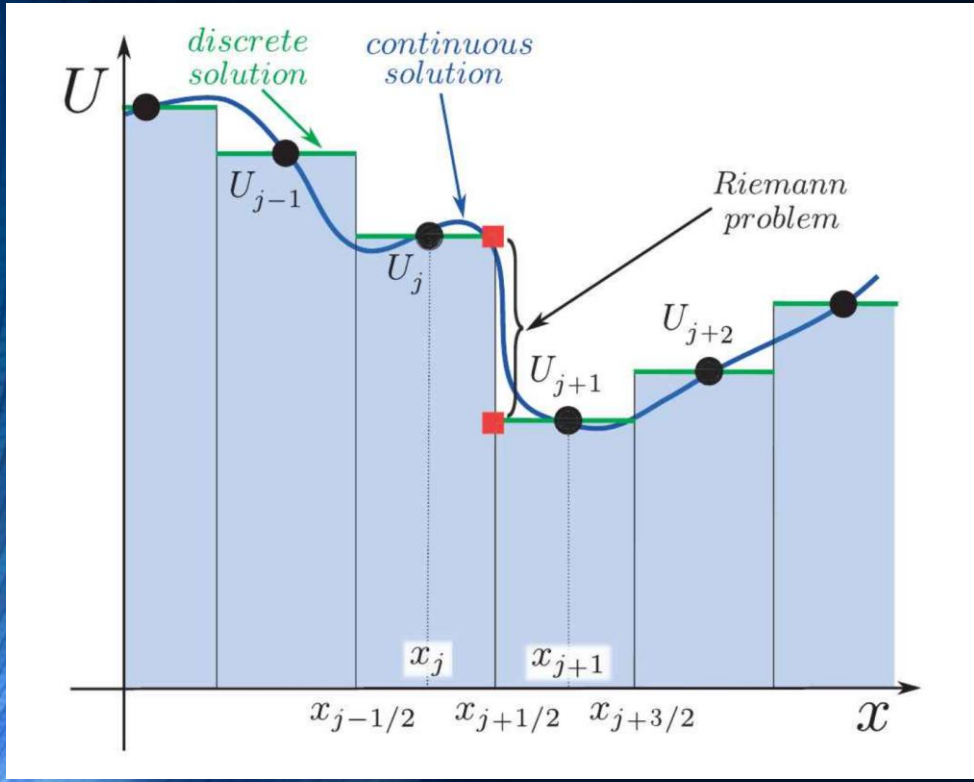
# The 3+1 Valencia Formulation of the Relativistic Hydrodynamic Equations

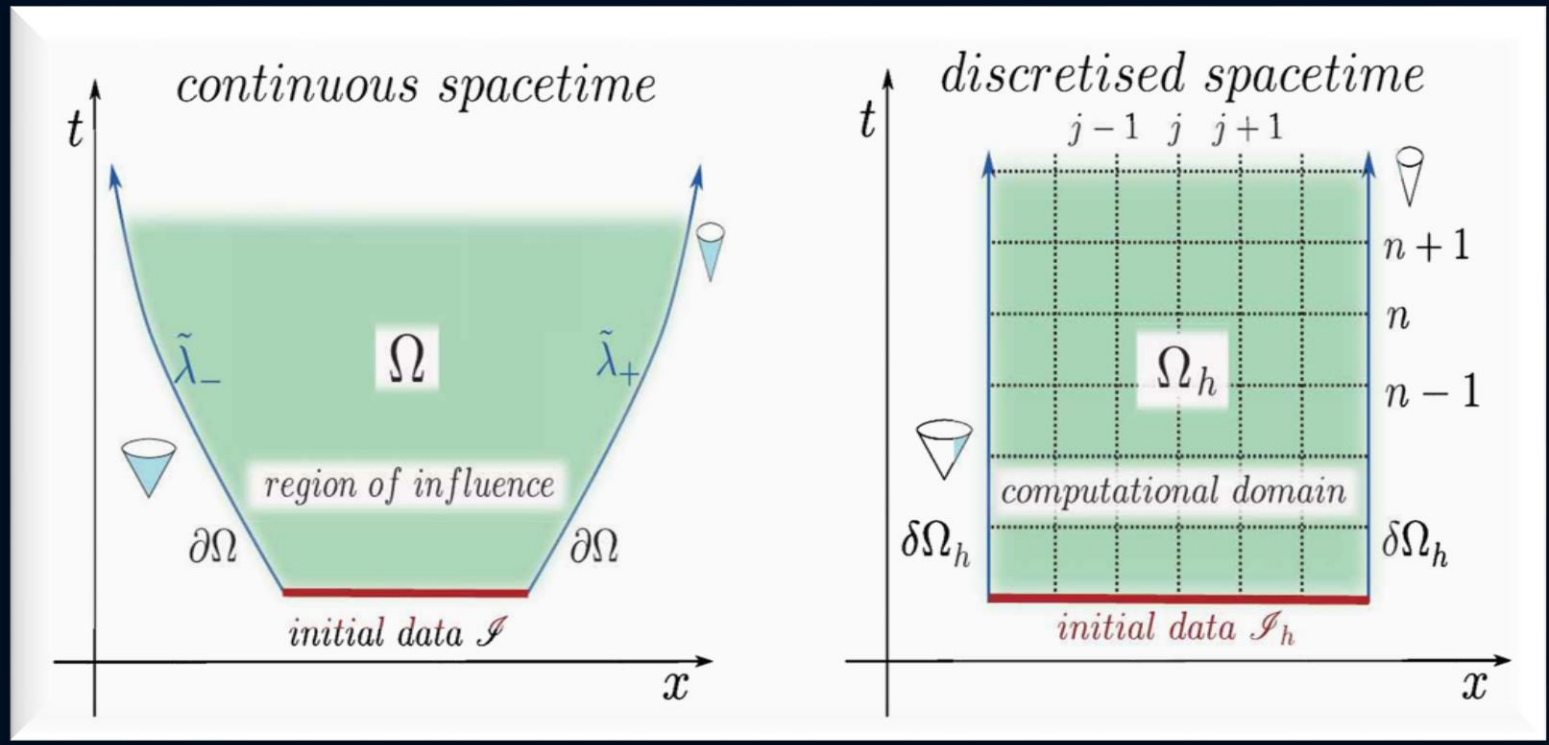
$$\nabla_\mu(\rho u^\mu) = 0\,,$$
$$\nabla_\nu T^{\mu\nu} = 0\,.$$

To guarantee that the numerical solution of the hydrodynamical equations (the conservation of rest mass and energy-momentum) converge to the right solution, they need to be reformulated into a conservative formulation. Most of the numerical “hydro codes” use here the 3+1 Valencia formulation.

# Finite difference methods

Discretisation of a hyperbolic initial value boundary problem.




High resolution shock capturing methods (HRSC methods) are needed, when Riemann problems of discontinuous properties and shocks needs to be evolved accurately.

All figures from: Luciano Rezzolla, Olindo Zanotti: Relativistic Hydrodynamics, Oxford Univ. Press, Oxford (2013)

# Gauge Conditions

On each spatial hypersurface, four additional degrees of freedom need to be specified: A slicing condition for the lapse function and a spatial shift condition for the shift vector need to be formulated to close the system. In an optimal gauge condition, singularities should be avoided and numerical calculations should be less time consuming.

Bona-Massó family of slicing conditions: $$\partial_t \alpha - \beta^k \partial_k \alpha = -f(\alpha)\,\alpha^2 (K - K_0)$$

"1+log" slicing condition: $$f = 2/\alpha \qquad \text{where } f(\alpha) > 0 \text{ and } K_0 := K(t=0)$$

"Gamma-Driver" shift condition:

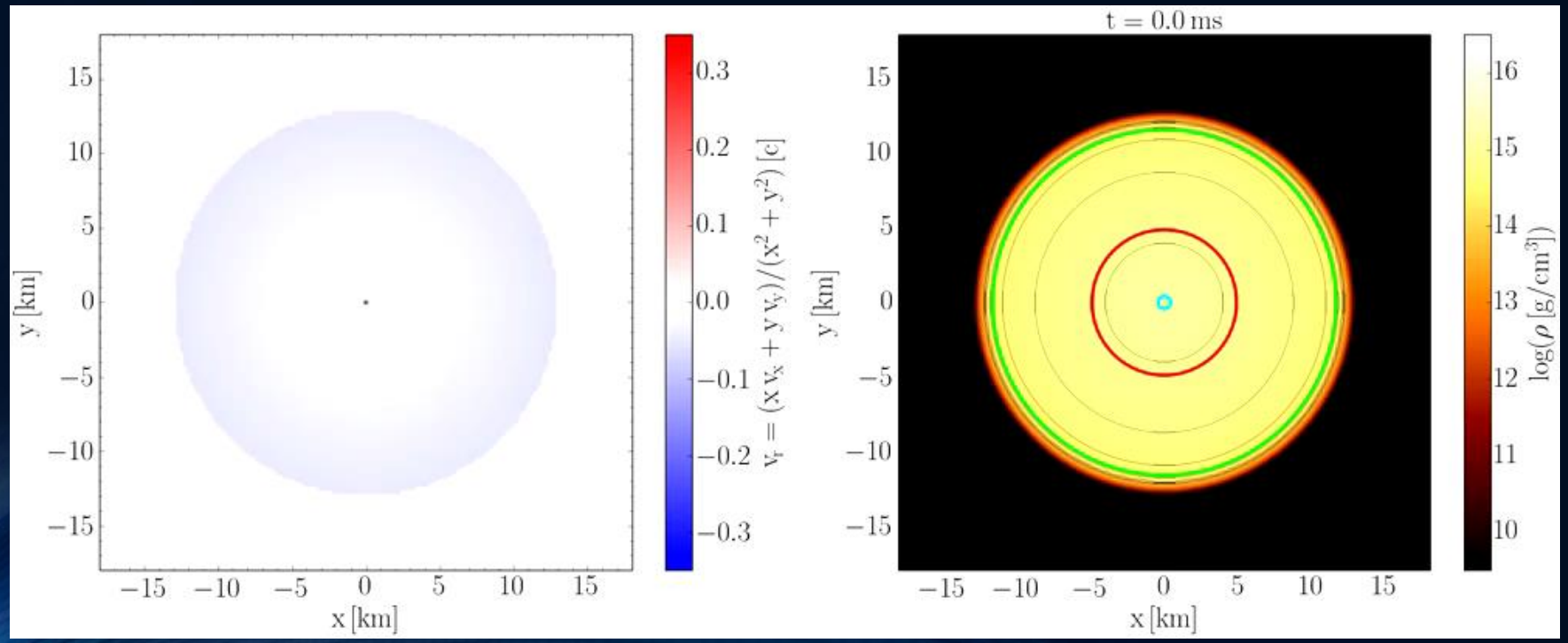
$$\partial_t \beta^i - \beta^j \partial_j \beta^i = \frac{3}{4} B^i,$$

$$\partial_t B^i - \beta^j \partial_j B^i = \partial_t \tilde{\Gamma}^i - \beta^j \partial_j \tilde{\Gamma}^i - \eta B^i$$

# Teil III

Inhalte des Teil III:

- How to download and build (compile) the Einstein Toolkit
- How to run a test simulation (static_tov.par)
- Run and visualize (Mathematica or Python) one of the following problems
  - Migration of an unstable neutron star to a stable configuration
  - Collapse of an unstable neutron star to a black hole
  - Collapse of a neutron star to a quark star (twin star collapse)

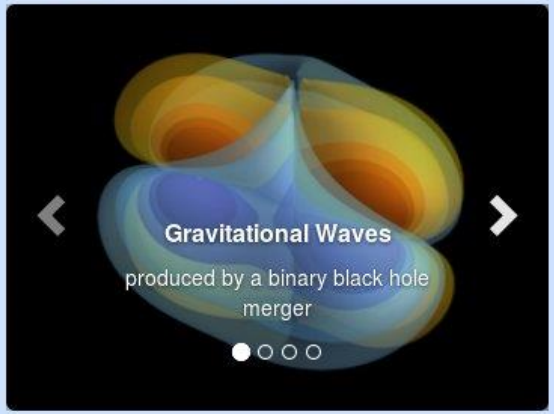


# Das Einstein Toolkit

einstein toolkit

Home About Download Documentation Help! Contribute Gallery

## The Einstein Toolkit




Gallery

### Einstein Toolkit School and Workshop

Join us at the North American Einstein Toolkit School and Workshop at NCSA, at the University of Illinois at Urbana-Champaign from July 31 to August 4 2017.

This meeting is open to anyone interested in numerical relativity and computational astrophysics and cosmology and in particular to Einstein toolkit users.

The first three days will be dedicated to a school useful for new users of the Einstein Toolkit followed by a two day long workshop open to developers interested in the Einstein Toolkit.

Registration closes *July 17, 2017*.

More information

### About

The Einstein Toolkit is a community-driven software platform of core computational tools to advance and support research in relativistic astrophysics and gravitational physics.

About

### Documentation

A lot of the documentation within the Einstein Toolkit is generated from comments in the source code, and more can be found on the Einstein Toolkit Wiki or other documents. We provide links to guides, tutorials and references.

Documentation

### Download

We provide a convenient method to get all of the Einstein Toolkit with just a few commands, and explain the whole process.

Download

### Contribute

The Einstein Toolkit would not exist without numerious contributions from its community. It is easy to learn how you can contribute as well.

Contribute

The Einstein Toolkit has been supported by NSF 1550551/1550461/1550436/1550514, NSF 1212401/1212426/1212433/1212460, NSF 0903973/0903782/0904015 (CIGR), 0701566/0855892 (XiRel), 0721915 (Alpaca), 0905046/0941653(PetaCactus/PRAC). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

# Das Einstein Toolkit: Download

einstein toolkit | Home | About | Download | Documentation | Help! | Contribute | Gallery

## Download & Requirements

The Einstein Toolkit is hosted on many different machines around the world. We provide a script called GetComponents to simplify downloading the toolkit. This page just describes how to download the toolkit - you may also be interested in the Tutorial for New Users which leads you through these steps and more on the Queen Bee supercomputer, or in a simpler tutorial for setup on a typical Linux box.

Users of the Einstein Toolkit are encouraged to register which also signs up for the users mailing list.

### Main Toolkit

#### Citations

The development of production level scientific software, such as the components of the Einstein Toolkit, represents the academic output of researchers. These scientific contributions should be acknowledged and respected on par with those solely based in theory or experiment. Please review our Citation Policy.

#### Current release: Payne-Gaposchkin (released on December 16th, 2016)

This is the recommended version of the toolkit for most users. See the release notes for more information.

Note: OSX users cannot use the 'subversion' client shipped by Apple. In that case install subversion either from homebrew or macports.

Enter the directory on your machine in which you would like to download the ET (for example, your home directory), and type the commands listed below. This will create a directory called Cactus in which the components of the Einstein Toolkit are downloaded.

```
curl -kLO https://raw.githubusercontent.com/gridaphobe/CRL/ET_2016_11/GetComponents
chmod a+x GetComponents
./GetComponents --parallel https://bitbucket.org/einsteintoolkit/manifest/raw/ET_2016_11/einsteintoolkit.th
```

A tarball of the release is also available here, but using GetComponents is the preferred method to obtain the code. Use the tarball only if there is no way to use GetComponents (which should almost never be the case).

# ET-Download auf dem Fuchs-Cluster

```
[prakti1@login02.csc ~]$ cd ET-2016-11/
[prakti1@login02.csc ET-2016-11]$ curl -kLO https://raw.githubusercontent.com/gridaphobe/CRL/ET_2016_11/GetComponents
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 99330  100 99330    0     0   486k      0 --:--:-- --:--:-- --:--:-- 30.9M
[prakti1@login02.csc ET-2016-11]$ chmod a+x GetComponents
[prakti1@login02.csc ET-2016-11]$ ./GetComponents --parallel https://bitbucket.org/einsteintoolkit/manifest/raw/ET_2016_11/einsteintoolkit.th
------------------------------------------------------------
  Checking out module: par
      from repository: https://bitbucket.org/einsteintoolkit/einsteinexamples.git
                 into: Cactus
```

```
                 into: Cactus
                   as: flesh
------------------------------------------------------------
  Checking out module: COPYRIGHT
      from repository: https://bitbucket.org/cactuscode/cactus.git
                 into: Cactus
                   as: flesh
------------------------------------------------------------
  Checking out module: doc
      from repository: https://bitbucket.org/cactuscode/cactus.git
                 into: Cactus
                   as: flesh
------------------------------------------------------------
  Checking out module: lib
      from repository: https://bitbucket.org/cactuscode/cactus.git
                 into: Cactus
                   as: flesh
------------------------------------------------------------
  Checking out module: ./utils
      from repository: https://bitbucket.org/cactuscode/utilities.git
                 into: Cactus
                   as: utils
------------------------------------------------------------
  Checking out module: Makefile
```

```
                 into: Cactus/arrangements
------------------------------------------------------------
  Checking out module: PITTNullCode/SphericalHarmonicDecomp
      from repository: https://bitbucket.org/einsteintoolkit/pittnullc
```

```
      from repository: https://bitbucket.org/einsteintoolkit/pittnullc
                 into: Cactus/arrangements
------------------------------------------------------------
  Checking out module: EinsteinInitialData/IDConstraintViolate
      from repository: https://bitbucket.org/einsteintoolkit/einsteini
                 into: Cactus/arrangements
------------------------------------------------------------
  Checking out module: ./CoreDoc
      from repository: https://bitbucket.org/cactuscode/coredoc.git
                 into: Cactus/arrangements/CactusDoc
                   as: CoreDoc
------------------------------------------------------------
  268 components checked out successfully.
  0 components updated successfully.

  Time Elapsed: 18 minutes, 5 seconds

[prakti1@login02.csc ET-2016-11]$
```

# Das Einstein Toolkit: Setup mit SimFactory

```
[prakti1@login02.csc Cactus]$ ./simfactory/bin/sim setup --machine fuchs


Here we will define some necessary Simulation Factory defaults.


Determining local machine name: login02.cm.cluster
Creating machine login02.cm.cluster from generic: machine login02.cm.cluster [/home/agmisc/prakti1/ET-2016-11/Cactus/repos/simfactory2/mdb/machin
enter value for key user [prakti1]:
enter value for key email [prakti1]:
enter value for key allocation []:
enter value for key sourcebasedir (the parent directory containing the Cactus sourcetree) [/home/agmisc/prakti1/ET-2016-11]:
enter value for key basedir (the location of simfactory simulations) [/home/agmisc/prakti1/simulations]:


would you like to enter key/value pairs for a specific machine? [Y/N*]:

--------------------SUMMARY--------------------:


[default]
user            = prakti1
email           = prakti1
allocation      =
sourcebasedir   = /home/agmisc/prakti1/ET-2016-11
basedir         = /home/agmisc/prakti1/simulations


------------------END SUMMARY------------------:

Save contents [Y*/N]:
Contents successfully written to /home/agmisc/prakti1/ET-2016-11/Cactus/repos/simfactory2/etc/defs.local.ini
[prakti1@login02.csc Cactus]$
```

# Das Einstein Toolkit: Kompilierung

```
[prakti1@login02.csc Cactus]$ ./simfactory/bin/sim build et --thornlist ./manifest/einsteintoolkit.th --machine fuchs
Using configuration: et
Reconfiguring et
Writing configuration to: /home/agmisc/prakti1/ET-2016-11/Cactus/configs/et/OptionList
Cactus - version: 4.2.3
Reconfiguring et.
Using configuration options from configure line
  Setting fds to '4,5 -j --'
End of options from configure line
Adding configuration options from '/home/agmisc/prakti1/ET-2016-11/Cactus/configs/et/OptionList'...
  Setting VERSION to '2015-05-16'
  Setting CPP to 'cpp'
  Setting FPP to 'cpp'
  Setting CC to '/cm/shared/apps/intel/composer_xe/2013_sp1.3.174/composer_xe_2013_sp1.3.174/bin/intel64/icc'
  Setting CXX to '/cm/shared/apps/intel/composer_xe/2013_sp1.3.174/composer_xe_2013_sp1.3.174/bin/intel64/icpc'
  Setting F77 to '/cm/shared/apps/intel/composer_xe/2013_sp1.3.174/composer_xe_2013_sp1.3.174/bin/intel64/ifort'
  Setting F90 to '/cm/shared/apps/intel/composer_xe/2013_sp1.3.174/composer_xe_2013_sp1.3.174/bin/intel64/ifort'
  Setting CPPFLAGS to '-DCCTK_DISABLE_OMP_COLLAPSE -DCCTK_DISABLE_RESTRICT'
  Setting FPPFLAGS to '-DCCTK_DISABLE_OMP_COLLAPSE -traditional -DCCTK_DISABLE_RESTRICT'
  Setting CFLAGS to '-g -traceback -msse3 -align -std=c99 -U__STRICT_ANSI__'
  Setting CXXFLAGS to '-g -traceback -msse3 -align -std=c++11 -D__builtin_fmaxf=fmaxf -D__builtin_fmaxl=fmaxl -D__builtin_fmi
  Setting F77FLAGS to '-g -traceback -msse3 -align -pad -safe-cray-ptr'
  Setting F90FLAGS to '-g -traceback -msse3 -align -pad -safe-cray-ptr'
  Setting C_LINE_DIRECTIVES to 'yes'
  Setting F_LINE_DIRECTIVES to 'yes'
  Setting LDFLAGS to '-Wl,--export-dynamic -Wl,-rpath,/cm/shared/apps/intel/composer_xe/2013_sp1.3.174/composer_xe_2013_sp1.3
th,/cm/shared/apps/intel/composer_xe/2013_sp1.3.174/composer_xe_2013_sp1.3.174/ipp/lib/intel64 -Wl,-rpath,/cm/shared/apps/int
_xe_2013_sp1.3.174/tbb/lib/intel64/gcc4.4'
  Setting BEGIN_WHOLE_ARCHIVE_FLAGS to '-Wl,--whole-archive'
```

# Das Einstein Toolkit: Weitere Informationen

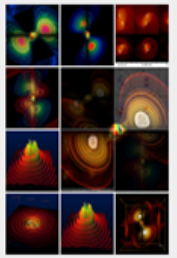
einstein toolkit

- Welcome
- About the Toolkit
  - Members
  - Maintainers
  - Governance
  - Capabilities
  - Gallery
  - Releases
  - Tools
- Download
- Community Services
  - Wiki
  - Blog
  - Support
  - Seminars
  - Issue Tracker
- Documentation
  - Tutorial for New Users
  - Citing

## Welcome

The Einstein Toolkit Consortium is developing and supporting open software for relativistic astrophysics. Our aim is to provide the core computational tools that can enable new science, broaden our community, facilitate interdisciplinary research and take advantage of emerging petascale computers and advanced cyberinfrastructure.

Please read our pages about the Einstein Toolkit, its governance, and how to get started with the toolkit for more information.



EinsteinToolkit@Flickr

## Download

**November 2014**: We are pleased to announce the tenth release (code name "Herschel") of the Einstein Toolkit, an open, community developed software infrastructure for relativistic astrophysics.

https://www.youtube.com/watch?v=EO4d32ch6OI
https://www.youtube.com/watch?v=p5bq2iUO3DE
https://www.youtube.com/watch?v=MNpyd_o0MT4
https://www.youtube.com/watch?v=Qg6PwRI2uS8
https://www.youtube.com/watch?v=ZW3aV7U-aik