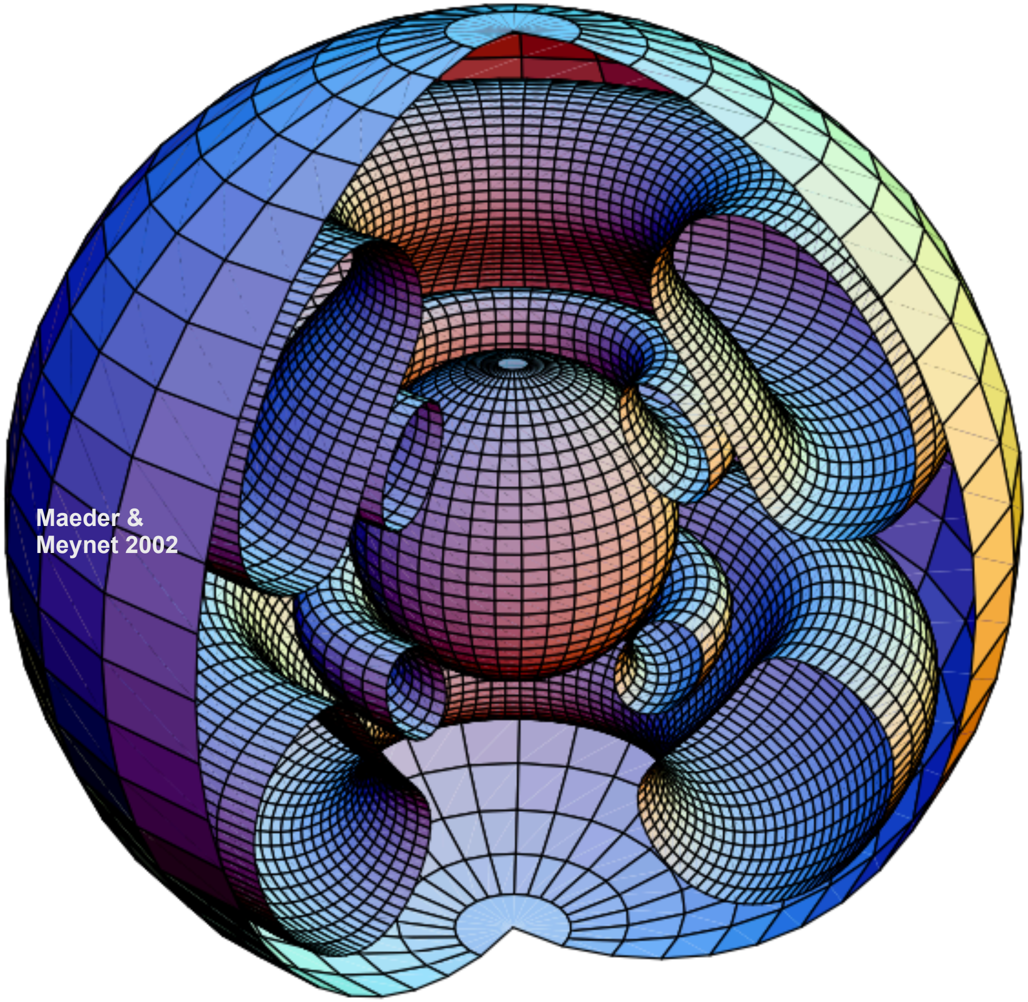
Stellar Rotation I



Matteo Cantiello
Kavli **I**nstitute for **T**heoretical **P**hysics
(University of California Santa Barbara)

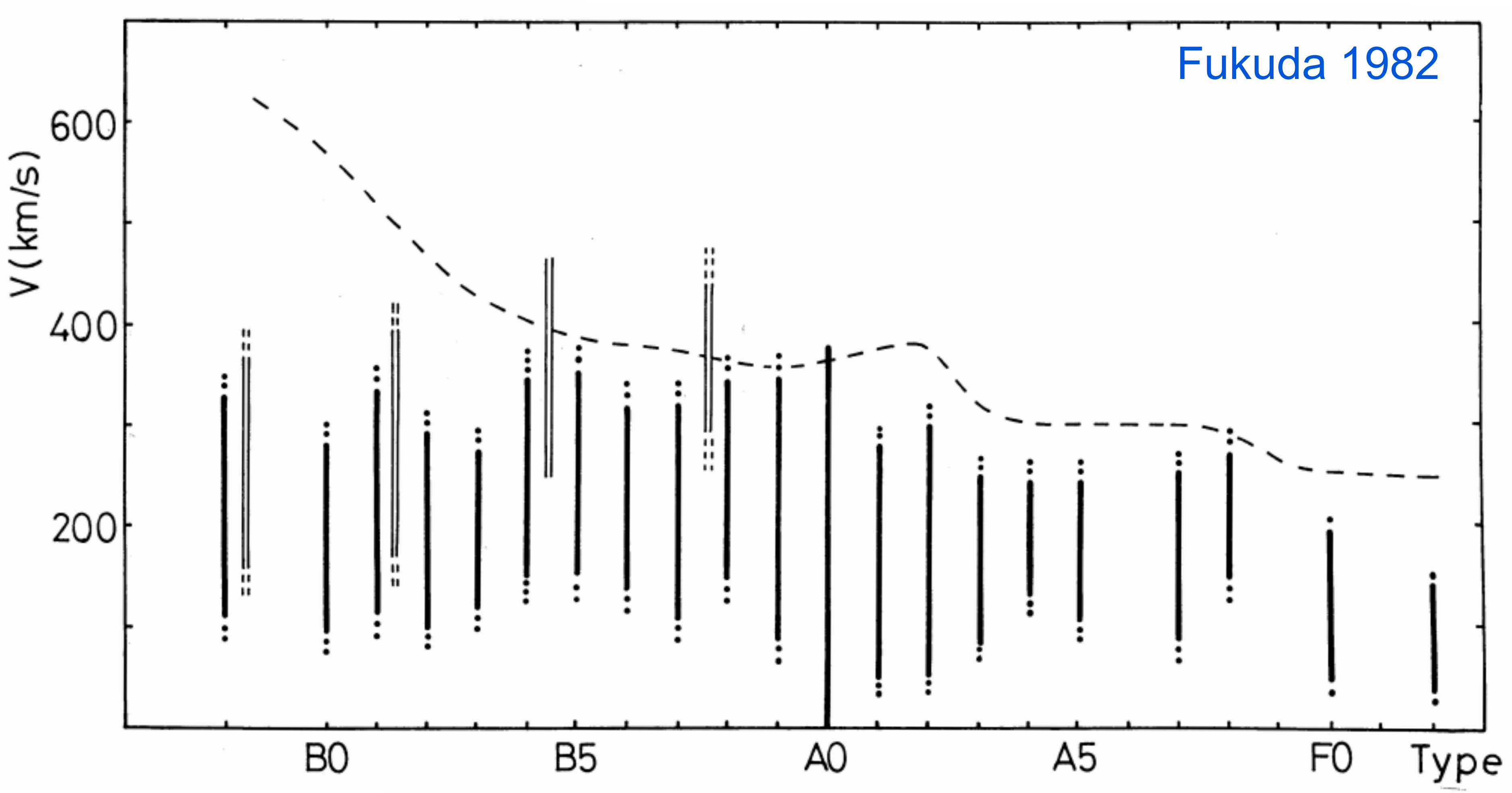
Outline

- Disclaimer
- Observed rotation rates
- Rotating stars
  - Hydrostatic equilibrium
  - Radiative equilibrium: Von Zeipel Theorem
- Modeling rotating stars in 1D: shellular rotation law
  - Diffusion Approximation
  - Meridional circulation
  - Rotational Instabilities
- Impact of rotation on stellar evolution
  - Lifetimes, Luminosities, Surface Abundances...
  - Final angular momentum content: SNe, GRBs
  - Spin of compact remnants (WD, NS, BH)

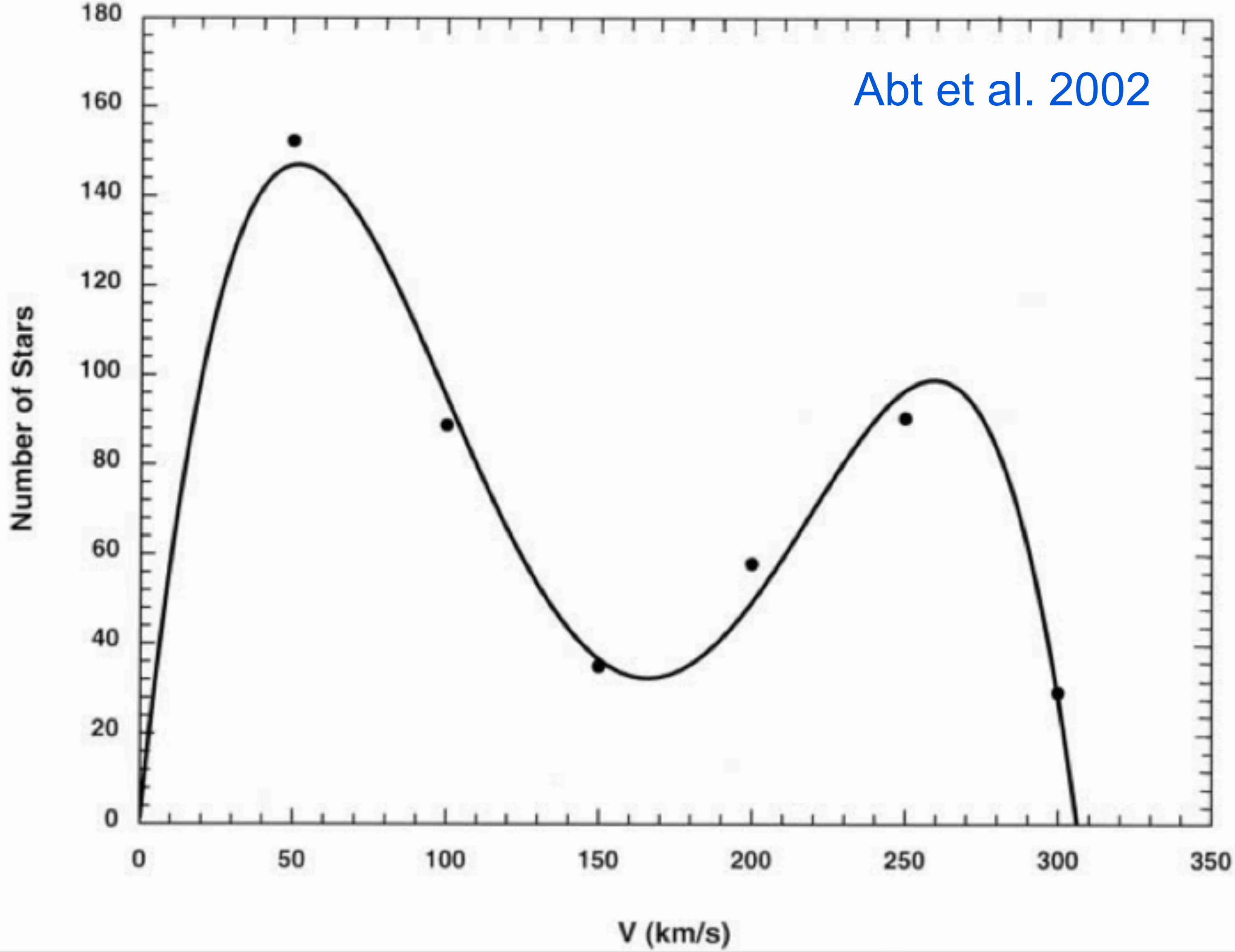
Rotating stars: observations

Range of rotational velocity

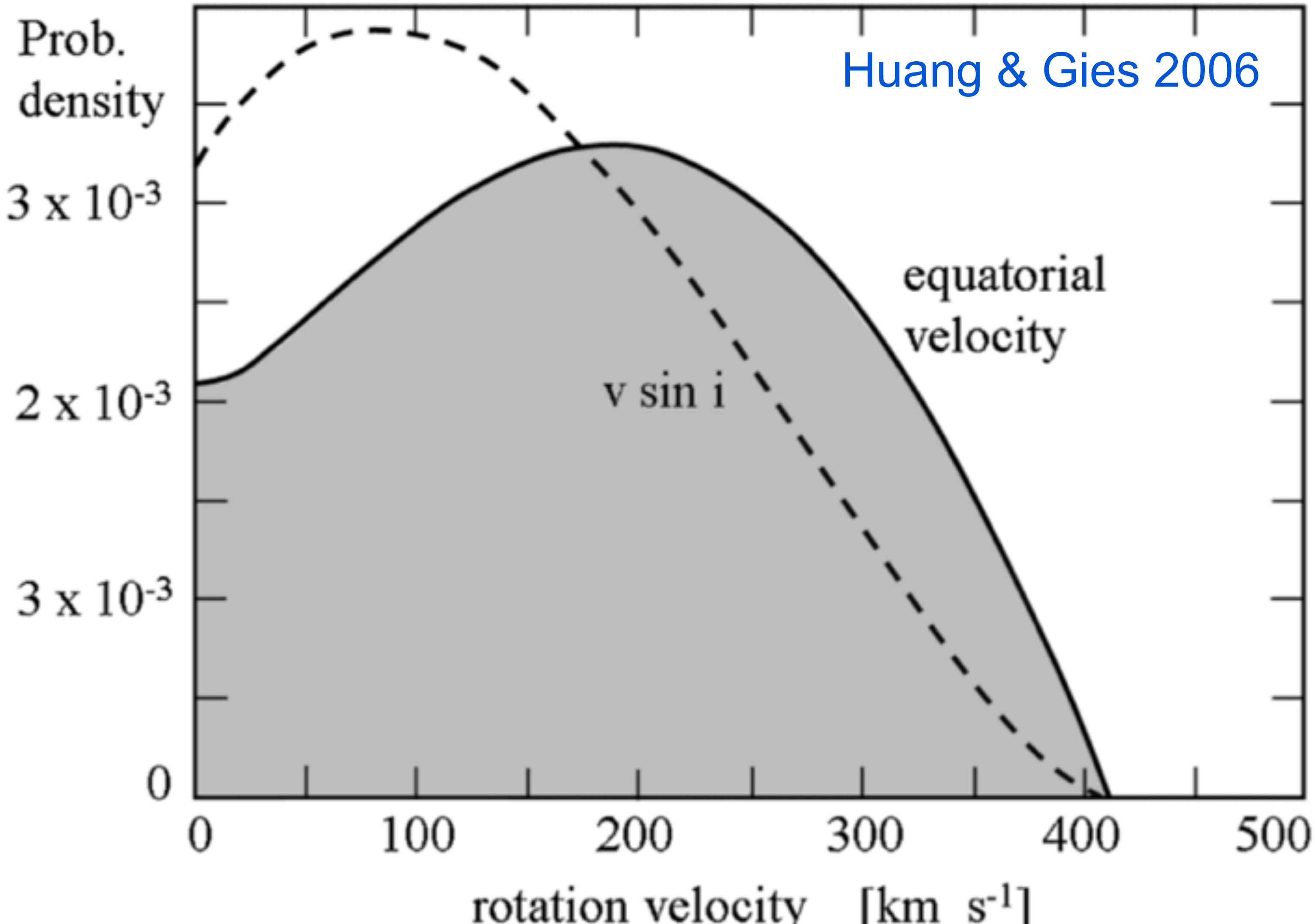
"In practice **all stars** are rotating around their axis" - Maeder & Meynet



B Stars in the bright star catalogue



Galactic OB Stars



VLT-Flames Tarantula Survey

VLT-FLAMES Tarantula Survey (PI: Evans): ~ 900 OB stars observed spectroscopically in 30 Dor (LMC) region.
Multi-epoch observations to separate binaries from single stars.

(Evans et al. 2010)

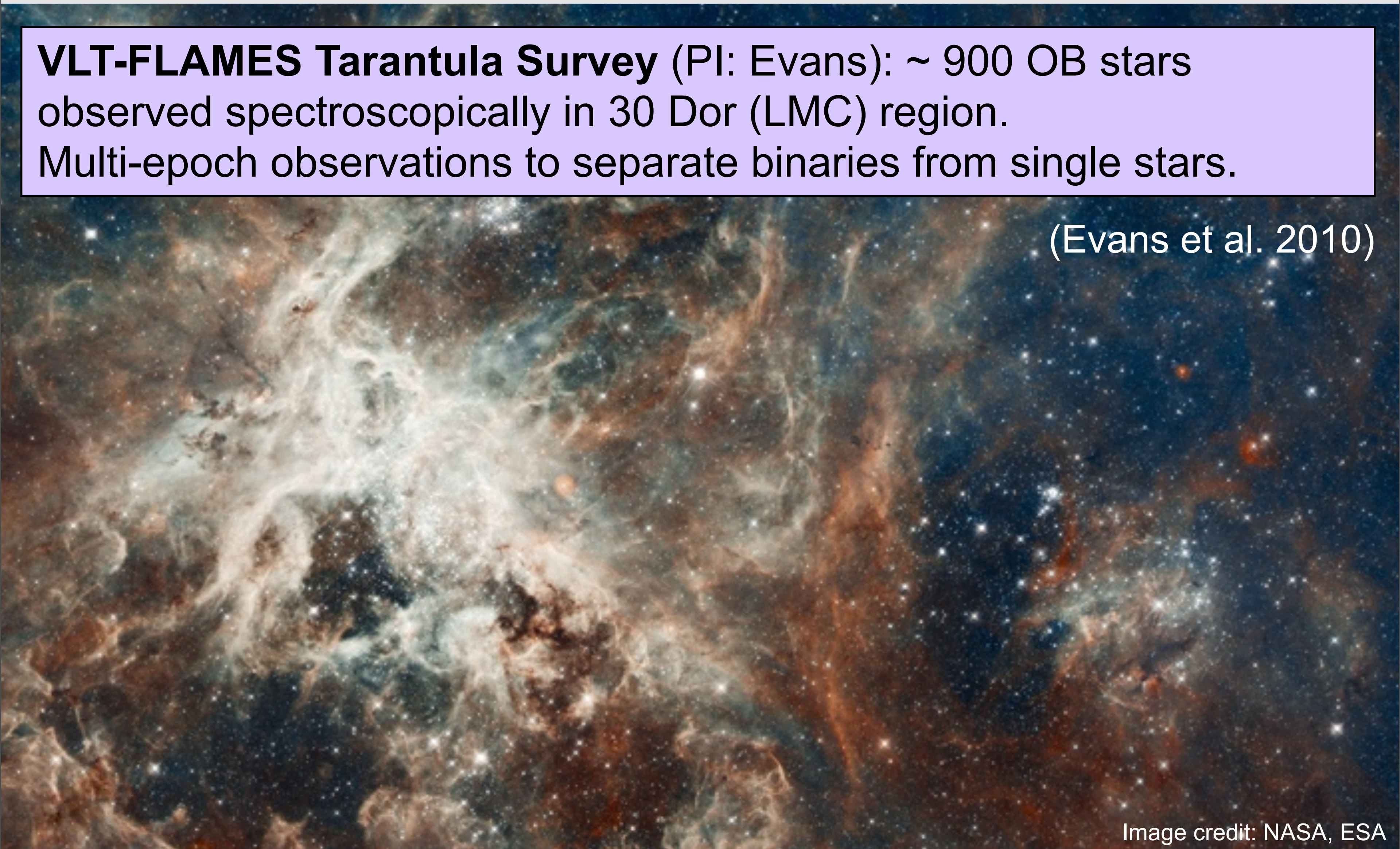
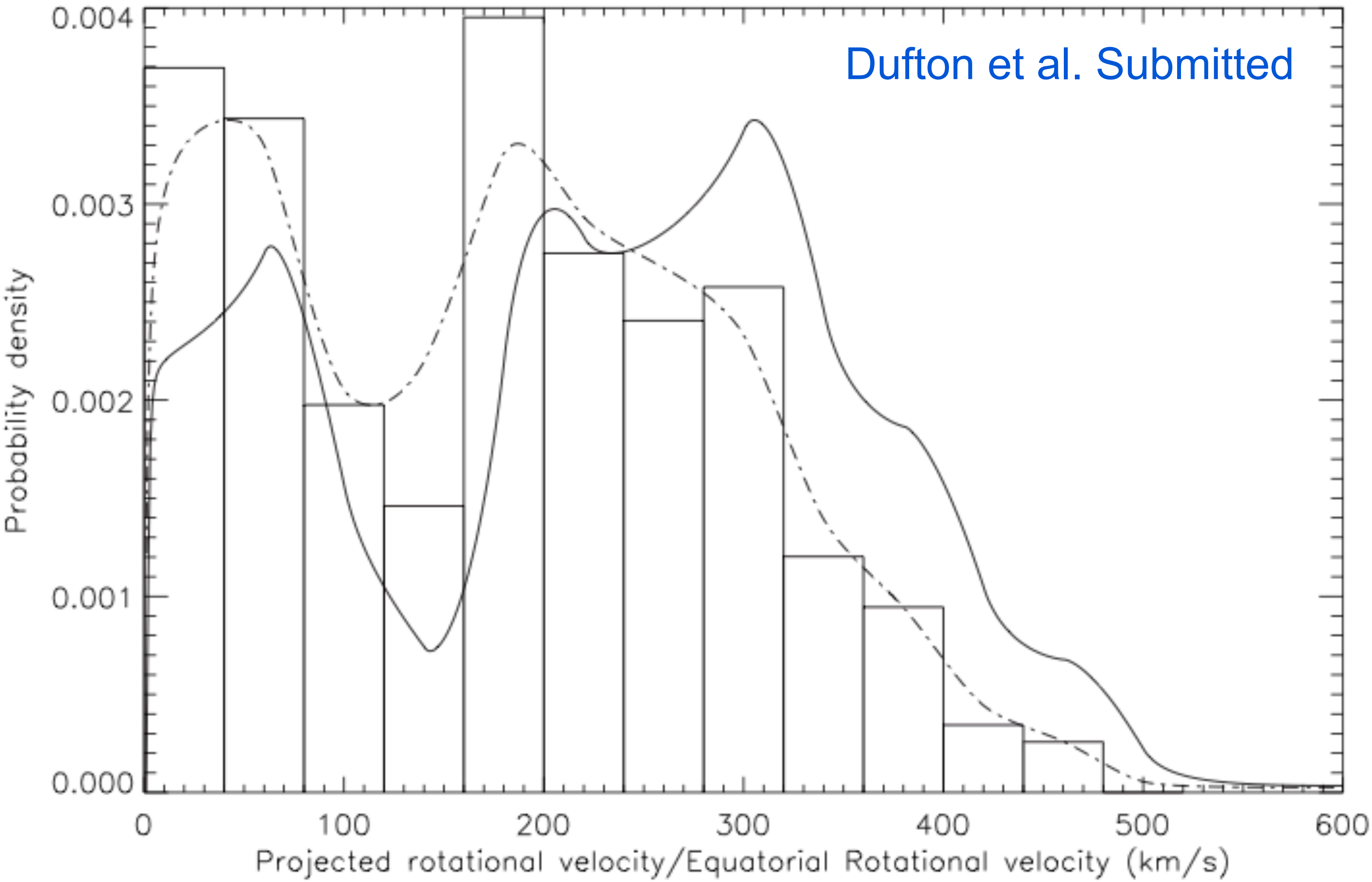
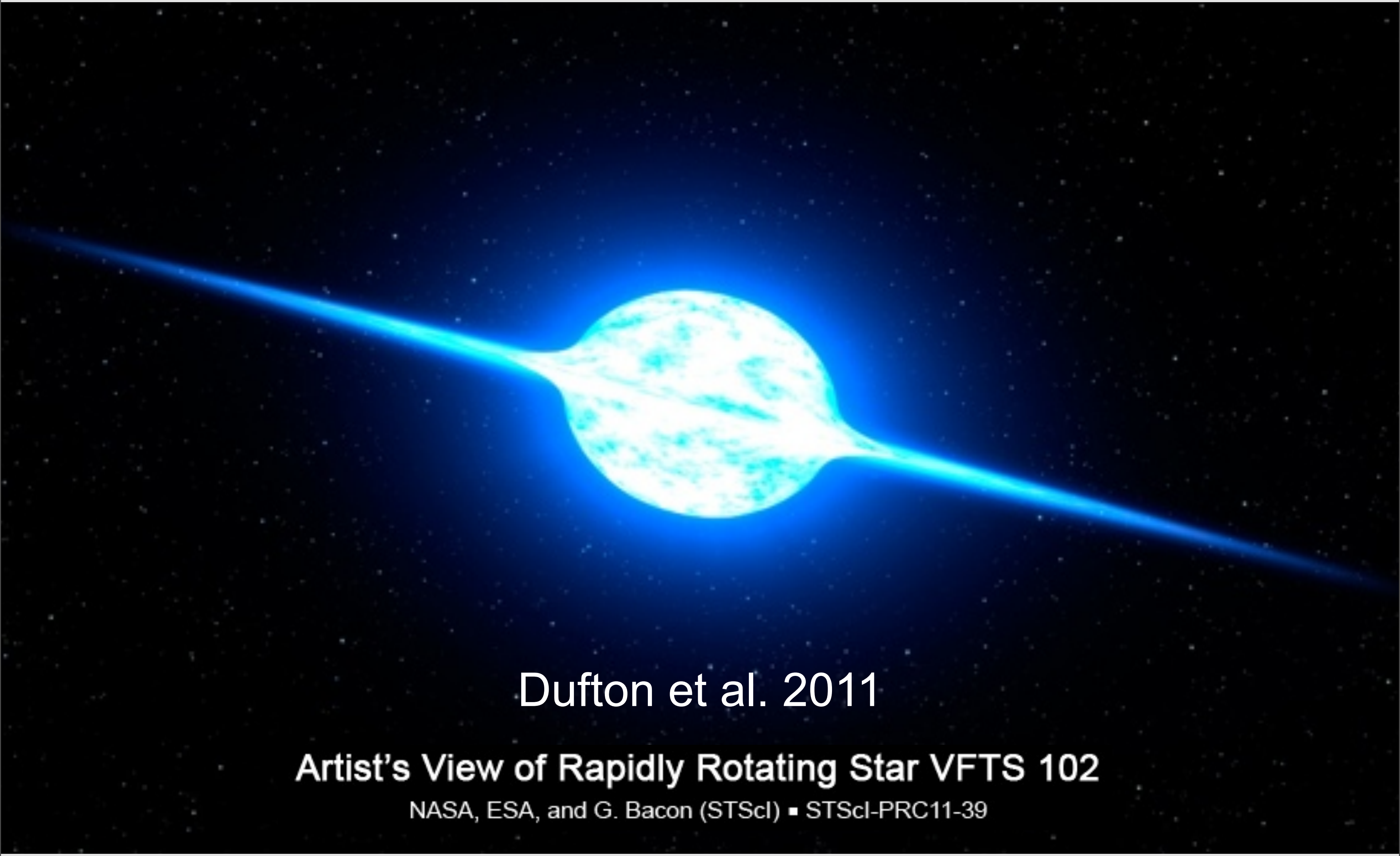


Image credit: NASA, ESA

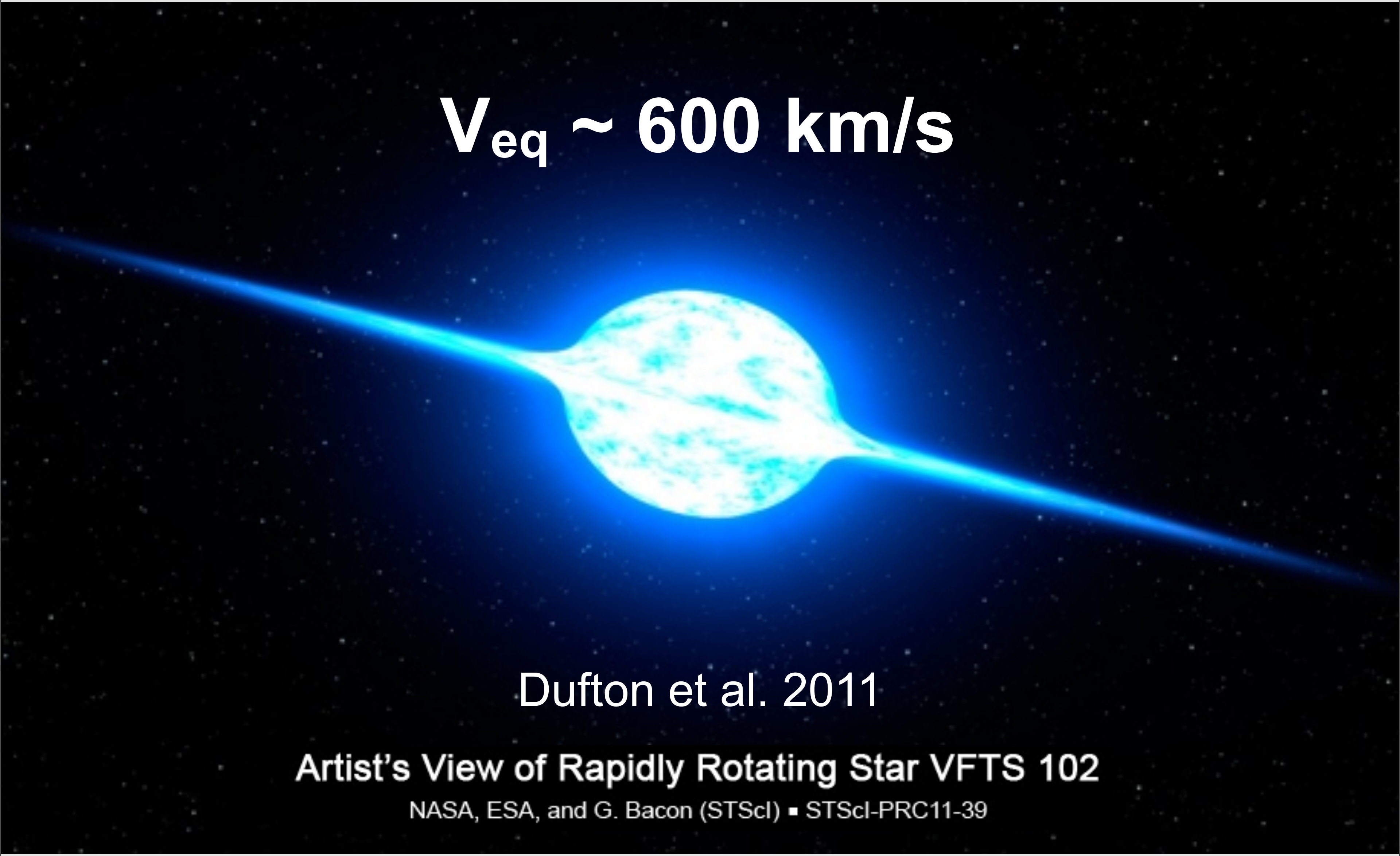
B Stars in 30 Doradus



VFTS 102, fastest rotating star*

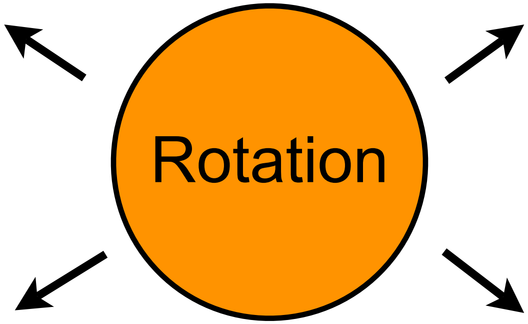


VFTS 102, fastest rotating star*

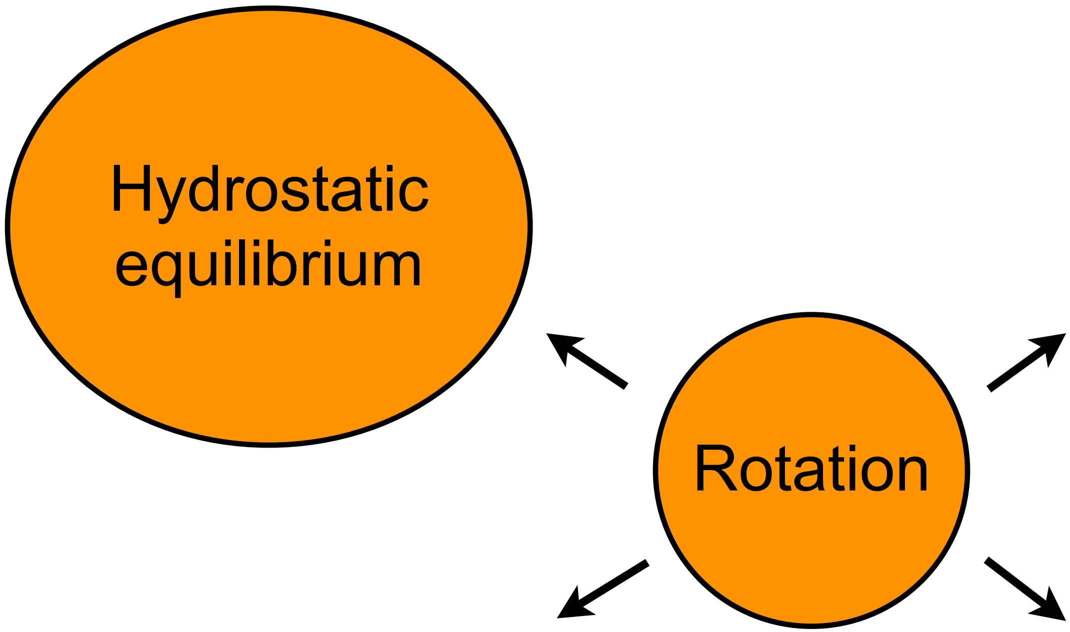


Effects of Rotation

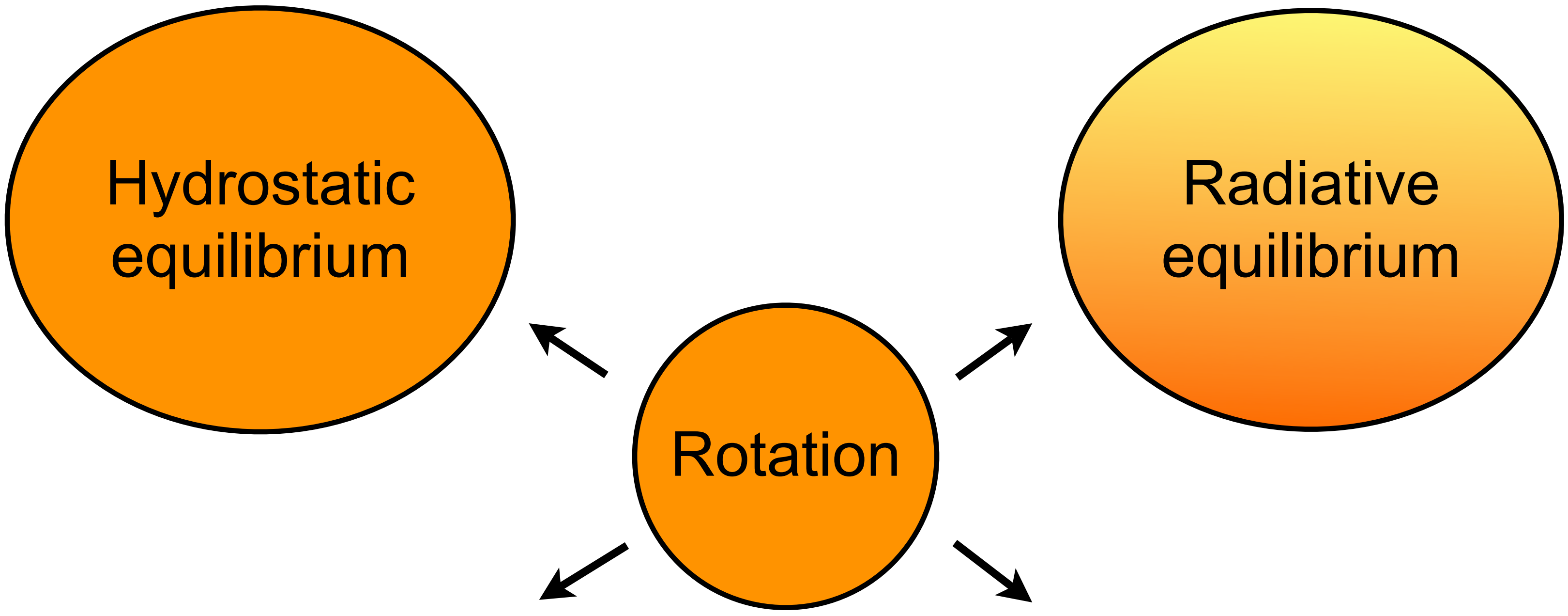
Effects of rotation



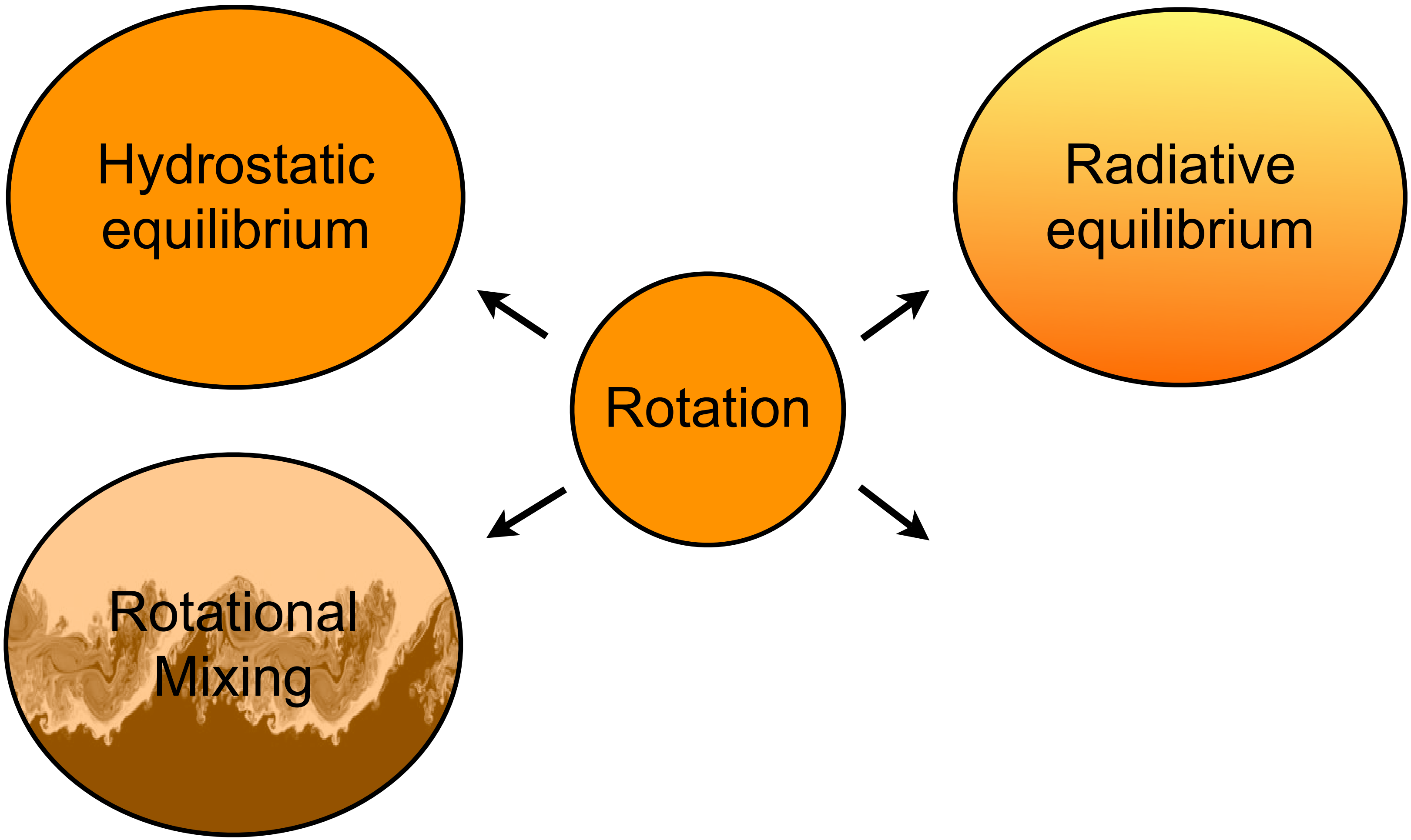
Effects of rotation



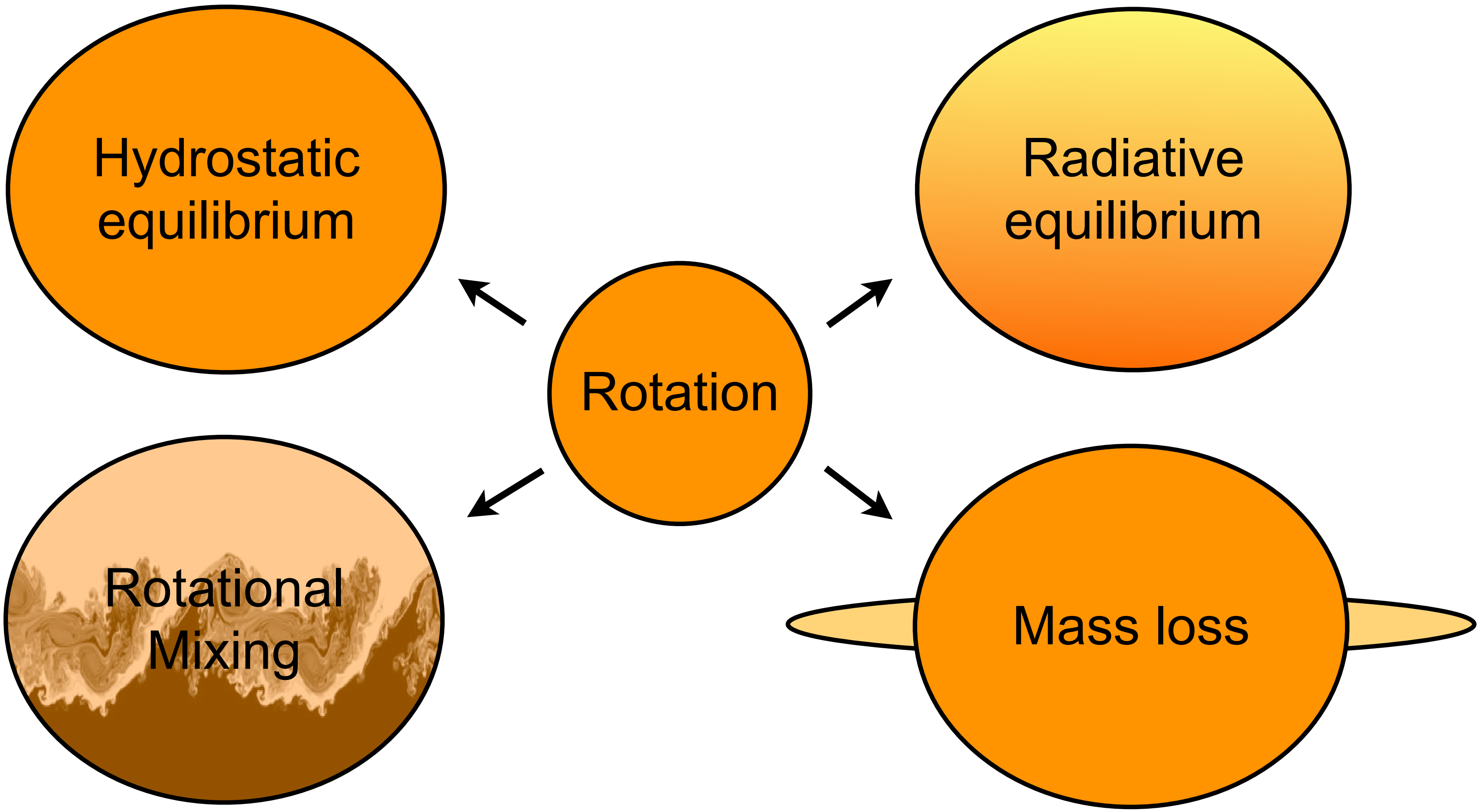
Effects of rotation



Effects of rotation

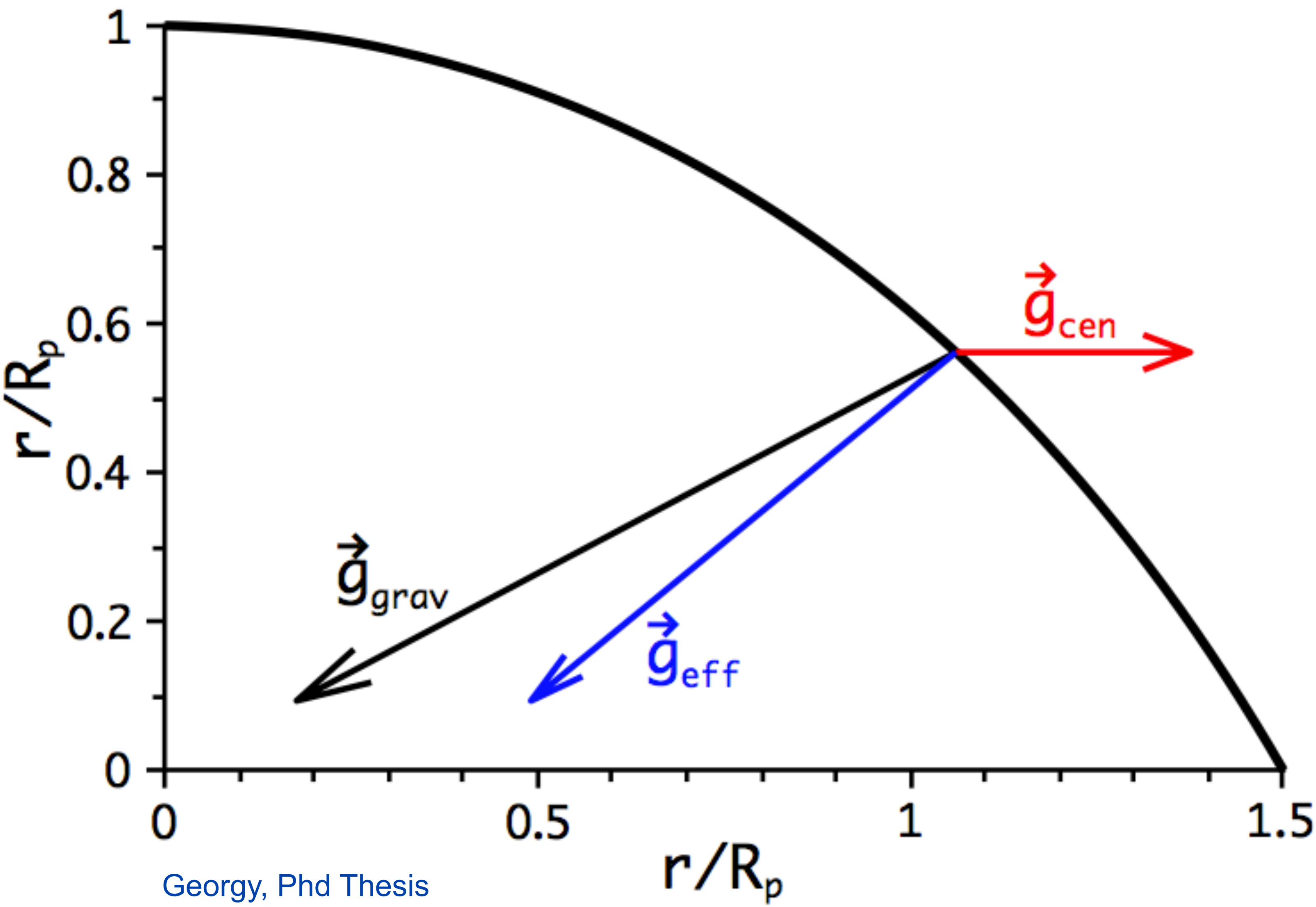


Effects of rotation



Hydrostatic Effects

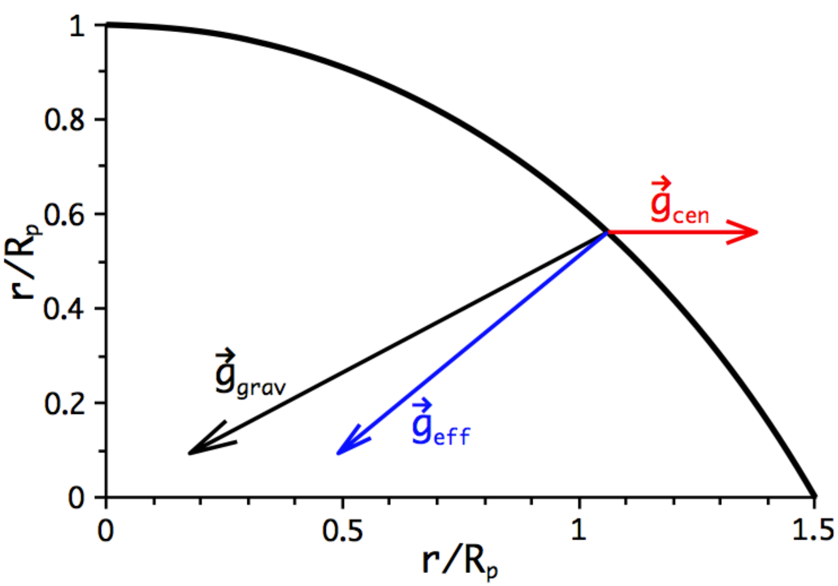
The centrifugal term



Georgy, Phd Thesis

The centrifugal term

The centrifugal force reduces the local gravity. The effect depends on the co-latitude, resulting in departures from spherical simmetry. Tassoul (1978) showed that, except for stars close to critical rotation, the deformation is axially symmetric. Even when this is not the case only the outer most layers of the star are affected (very little mass).



In the **Roche** approximation:

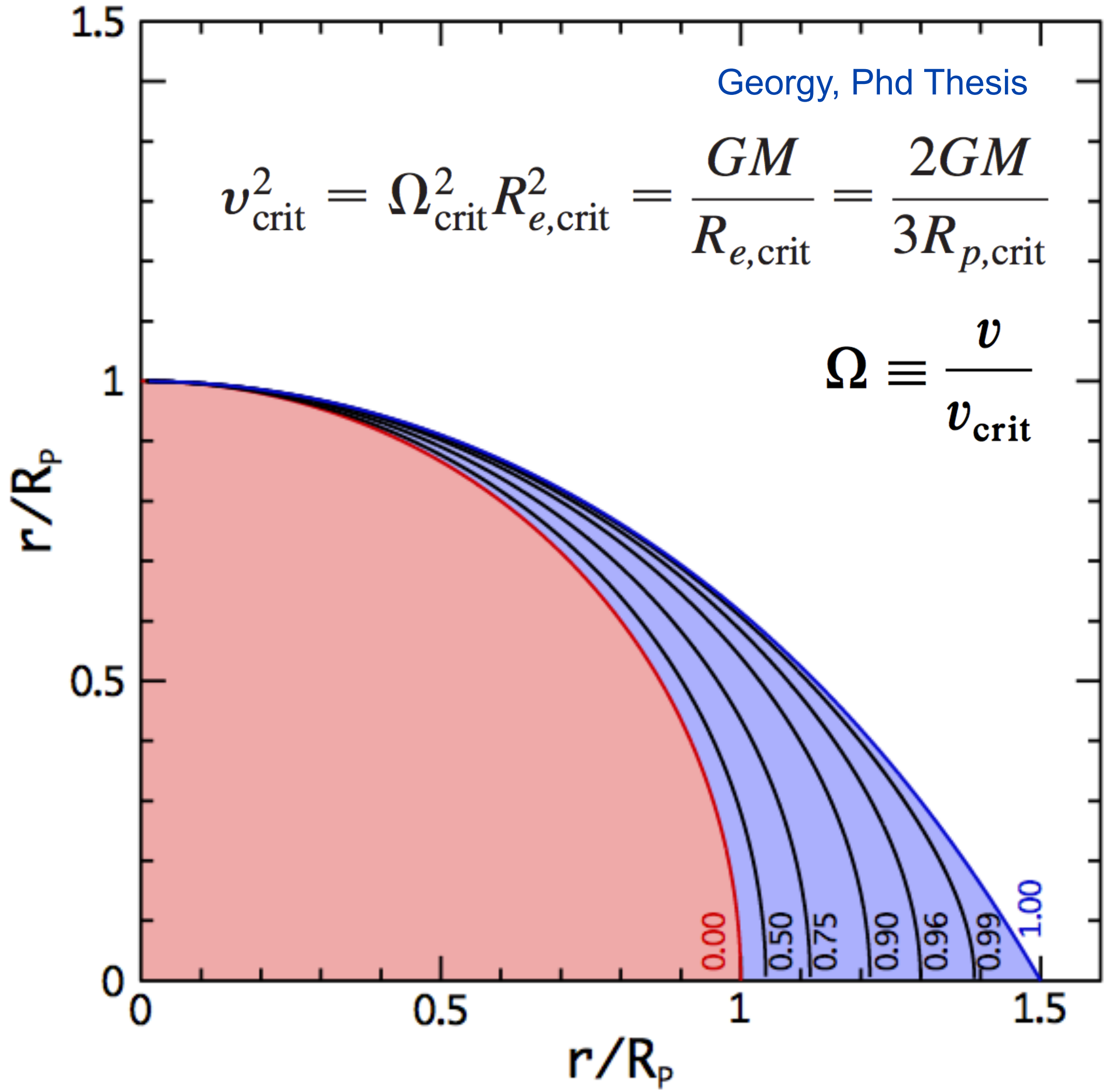
$$\vec{g}_{\text{eff}} = \left(-\frac{GM}{r^2} + \Omega^2 r \sin^2(\theta)\right)\vec{e}_r + \Omega^2 r \sin(\theta)\cos(\theta)\vec{e}_\theta$$

$$= \left(-\frac{\partial \Phi}{\partial r} + \Omega^2 r \sin^2(\theta)\right)\vec{e}_r + \Omega^2 r \sin(\theta)\cos(\theta)\vec{e}_\theta,$$

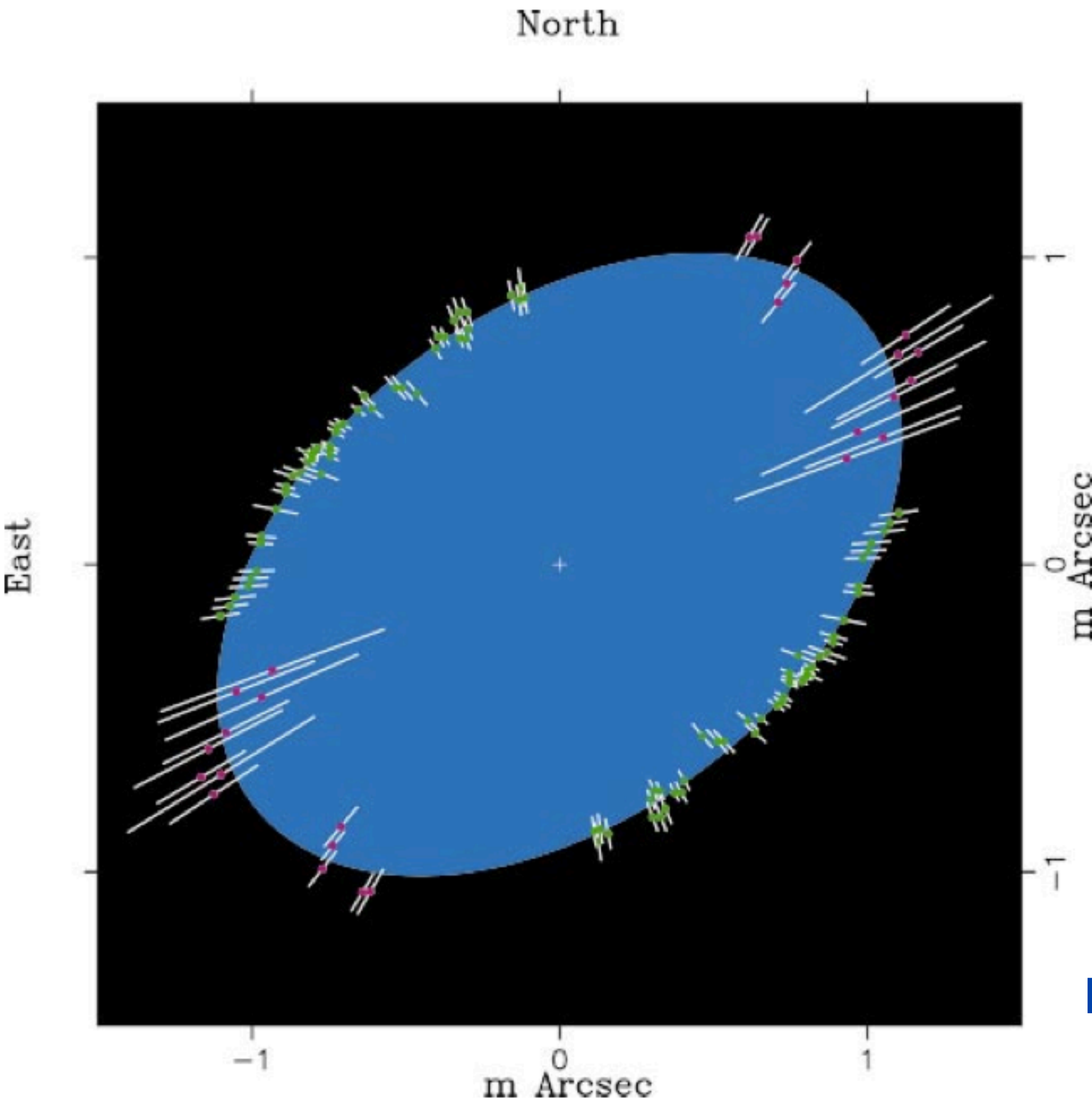
Departure from spherical structure

In the **Roche** approximation the deformation of the star approaches the value

R_{eq}/R_p=1.5



Interferometry of rotating stars



Achernar (Be star)
Star rotating at 96% of critical velocity (311 km/s)

VLTI Observations show that:

R_{eq}/R_{p}~1.5

Domiciano Da Souza et al.2003

Barotropic Star

$$\frac{1}{\varrho}\vec{\nabla}P = -\vec{\nabla}\Phi + \frac{1}{2}\Omega^2\vec{\nabla}(r\sin\vartheta)^2$$

$$\Psi = \Phi + V$$

$$\frac{1}{\varrho}\vec{\nabla}P = -\vec{\nabla}\Psi = \vec{g}_{\text{eff}}$$

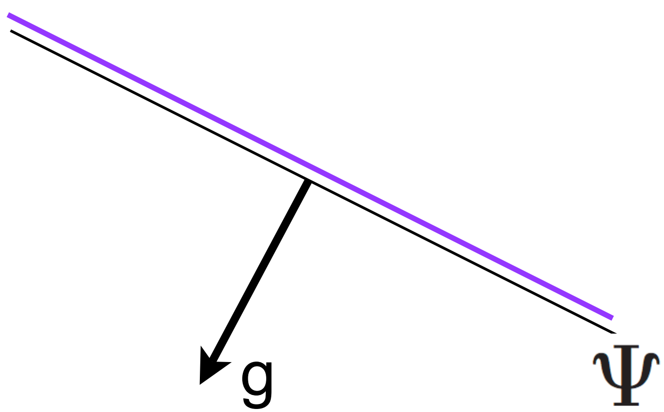
If Omega is constant (Solid body rotation) or has cilindric symmetry, the centrifugal acceleration can be derived from a potential (V). The eq. of Hydrostatic Equilibrium then implies that the star is **Barotropic**

Barotropic Star

$$\frac{1}{\varrho}\vec{\nabla}P = -\vec{\nabla}\Phi + \frac{1}{2}\Omega^2\vec{\nabla}(r\sin\vartheta)^2$$

$$\Psi = \Phi + V$$

$$\frac{1}{\varrho}\vec{\nabla}P = -\vec{\nabla}\Psi = \vec{g}_{\text{eff}} \longrightarrow \begin{aligned} P &= P(\Psi) \\ T &= T(\Psi) \end{aligned}$$



$$\varrho, P, T$$

<u>constant on equipotentials</u>

Baroclinic Star

$$\frac{1}{\varrho}\vec{\nabla}P = -\vec{\nabla}\Phi + \frac{1}{2}\Omega^2\vec{\nabla}(r\sin\vartheta)^2$$

$$\Psi = \Phi + V$$

$$\frac{1}{\varrho}\vec{\nabla}P = -\vec{\nabla}\Psi = \vec{g}_{\mathrm{eff}}$$

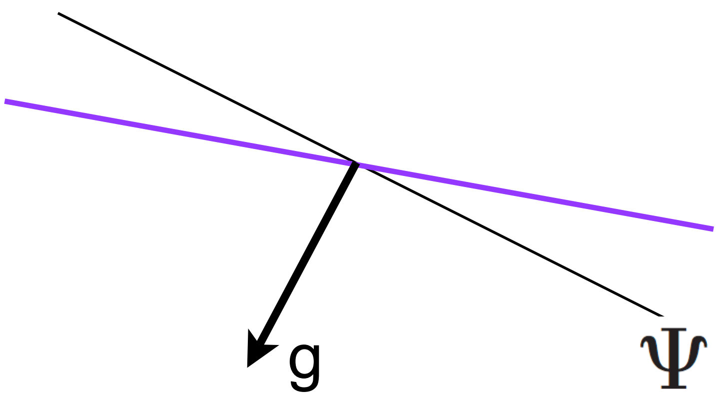
For different rotation laws (e.g. Shellular), the centrifugal acceleration can not be derived from a potential (V).
In this case Isobars and Equipotentials DO NOT coincide.
The star is **Baroclinic**

Baroclinic Star

$$\frac{1}{\varrho}\vec{\nabla}P = -\vec{\nabla}\Phi + \frac{1}{2}\Omega^2\vec{\nabla}(r\sin\vartheta)^2$$

$$\Psi \neq \Phi + V$$

$$\frac{1}{\varrho}\vec{\nabla}P \neq -\vec{\nabla}\Psi = \vec{g}_{\rm eff} \quad \longrightarrow \quad \begin{array}{l} P \neq P(\Psi) \\ T \neq T(\Psi) \end{array}$$

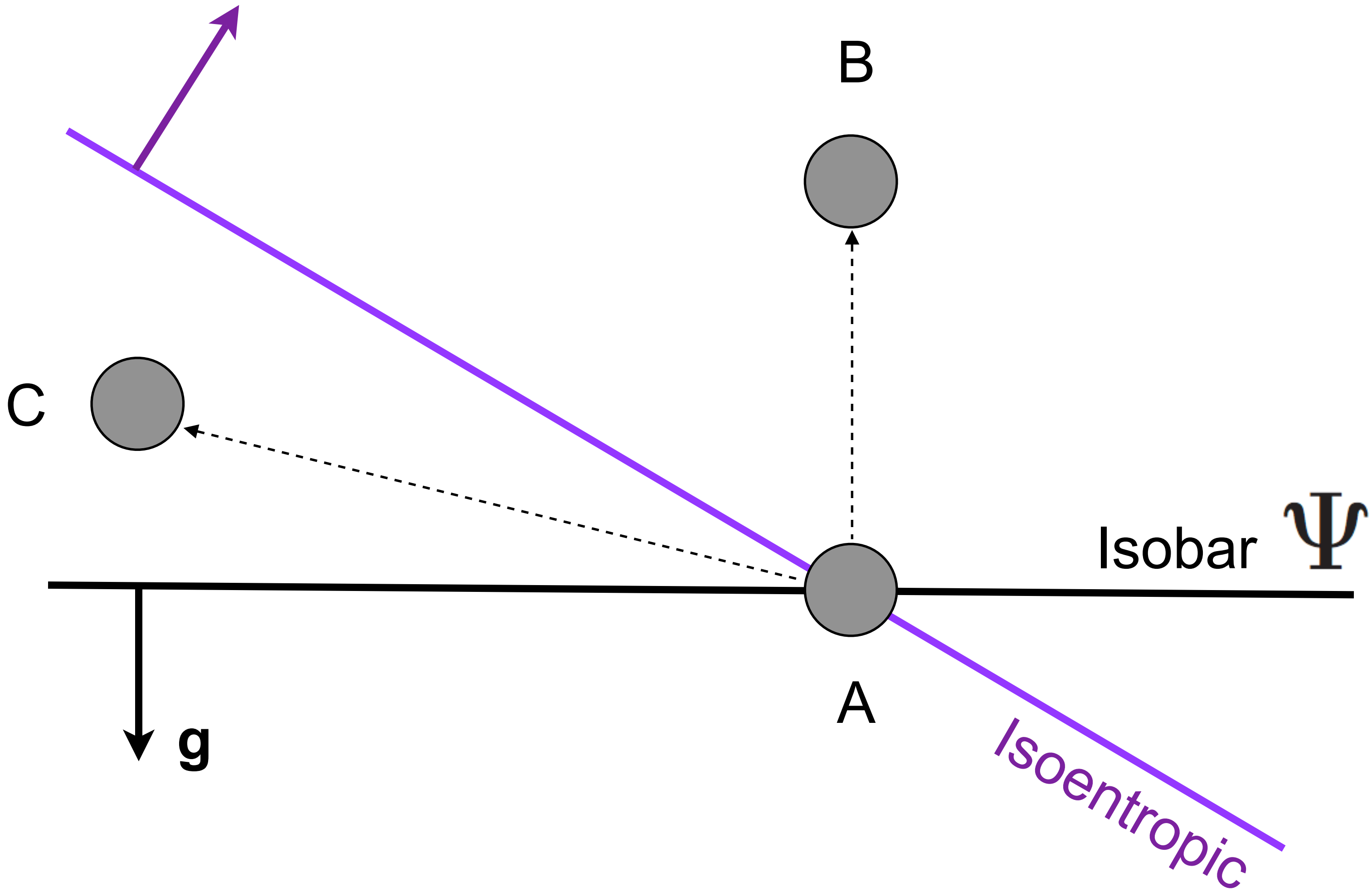


$$\longleftarrow \quad \varrho, P, T$$

<u>not constant on equipotentials</u>

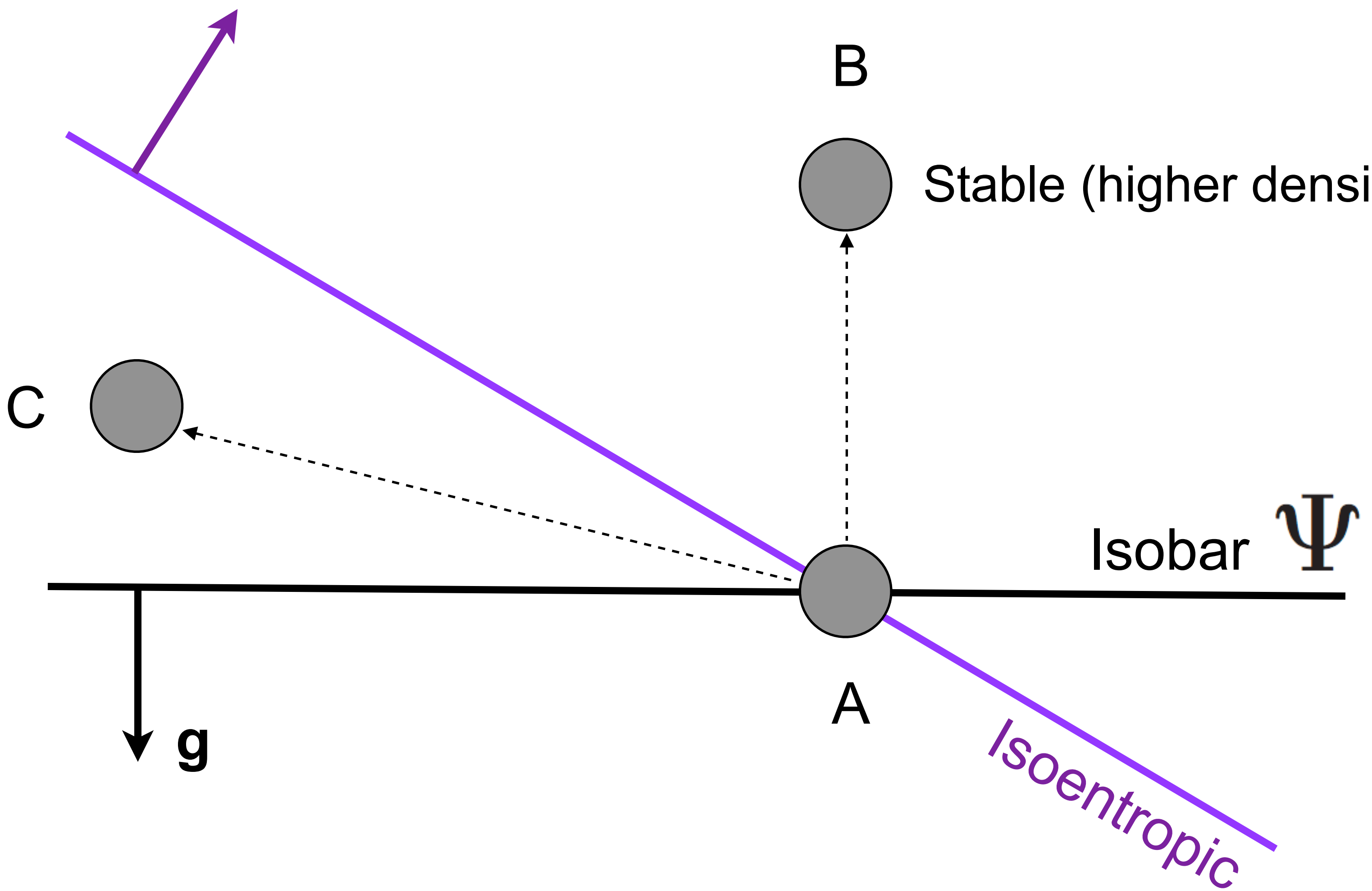
Baroclinicity leads to instabilities

Assumption: adiabatic displacement



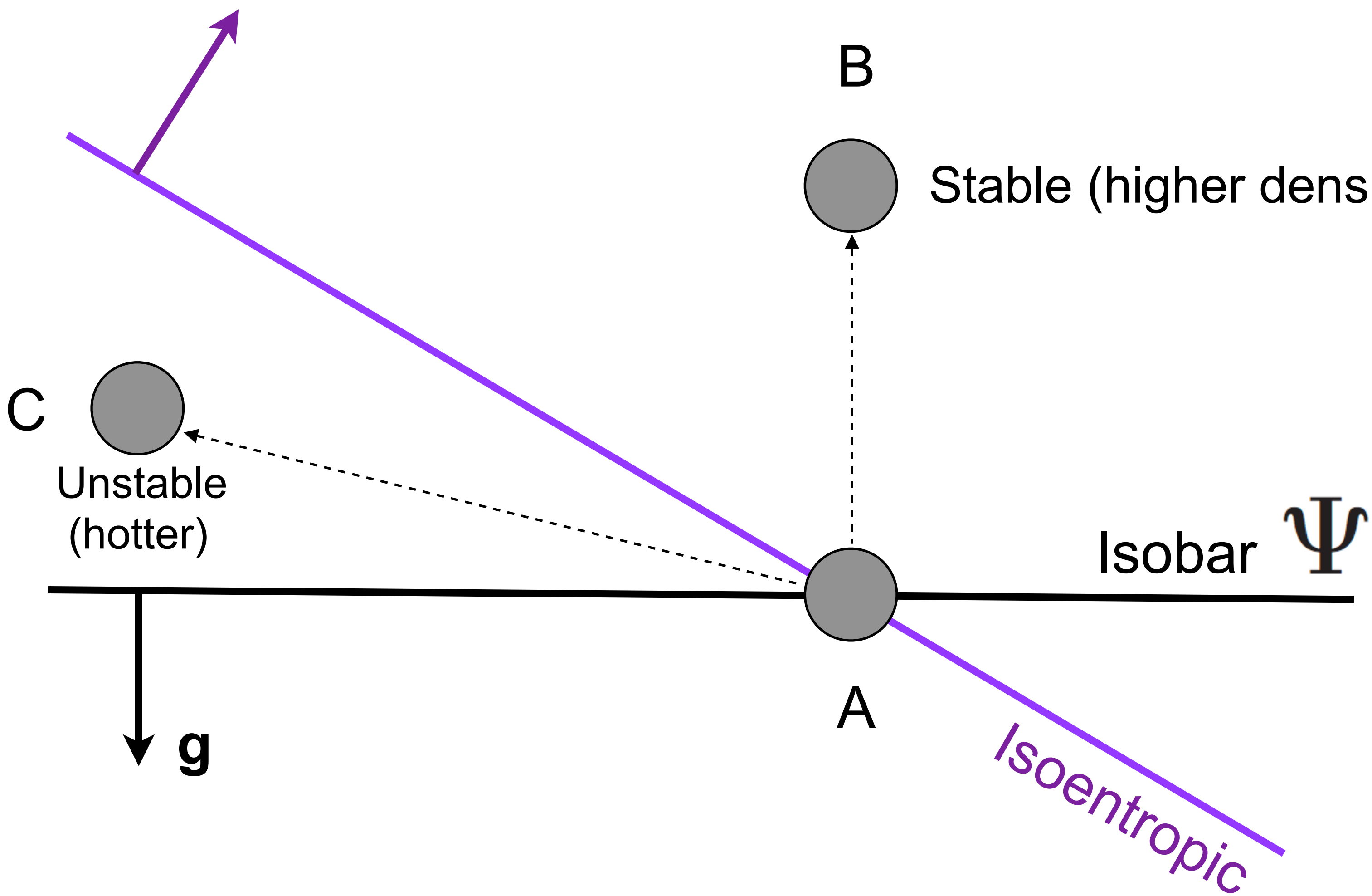
Baroclinicity leads to instabilities

Assumption: adiabatic displacement

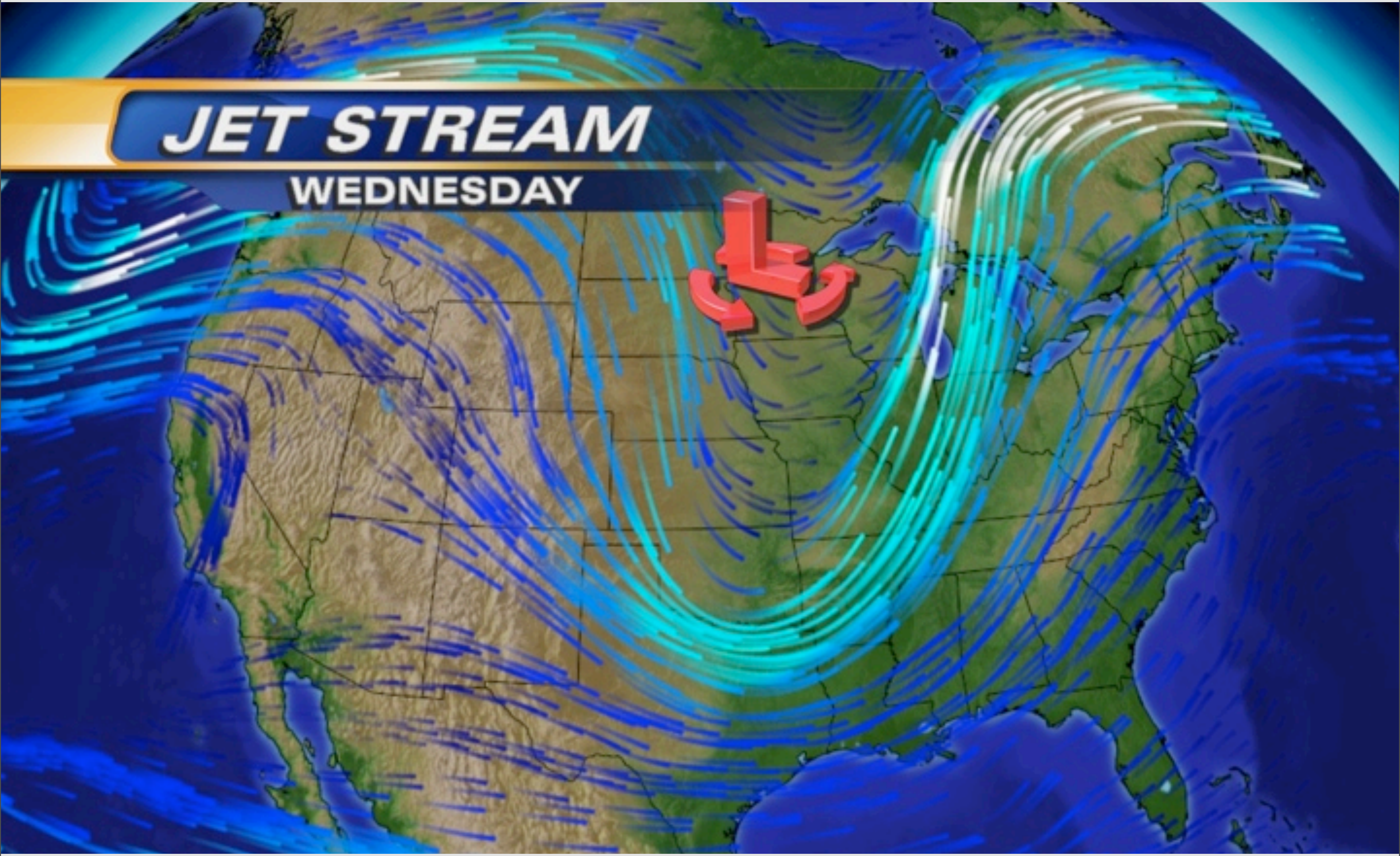


Baroclinicity leads to instabilities

Assumption: adiabatic displacement

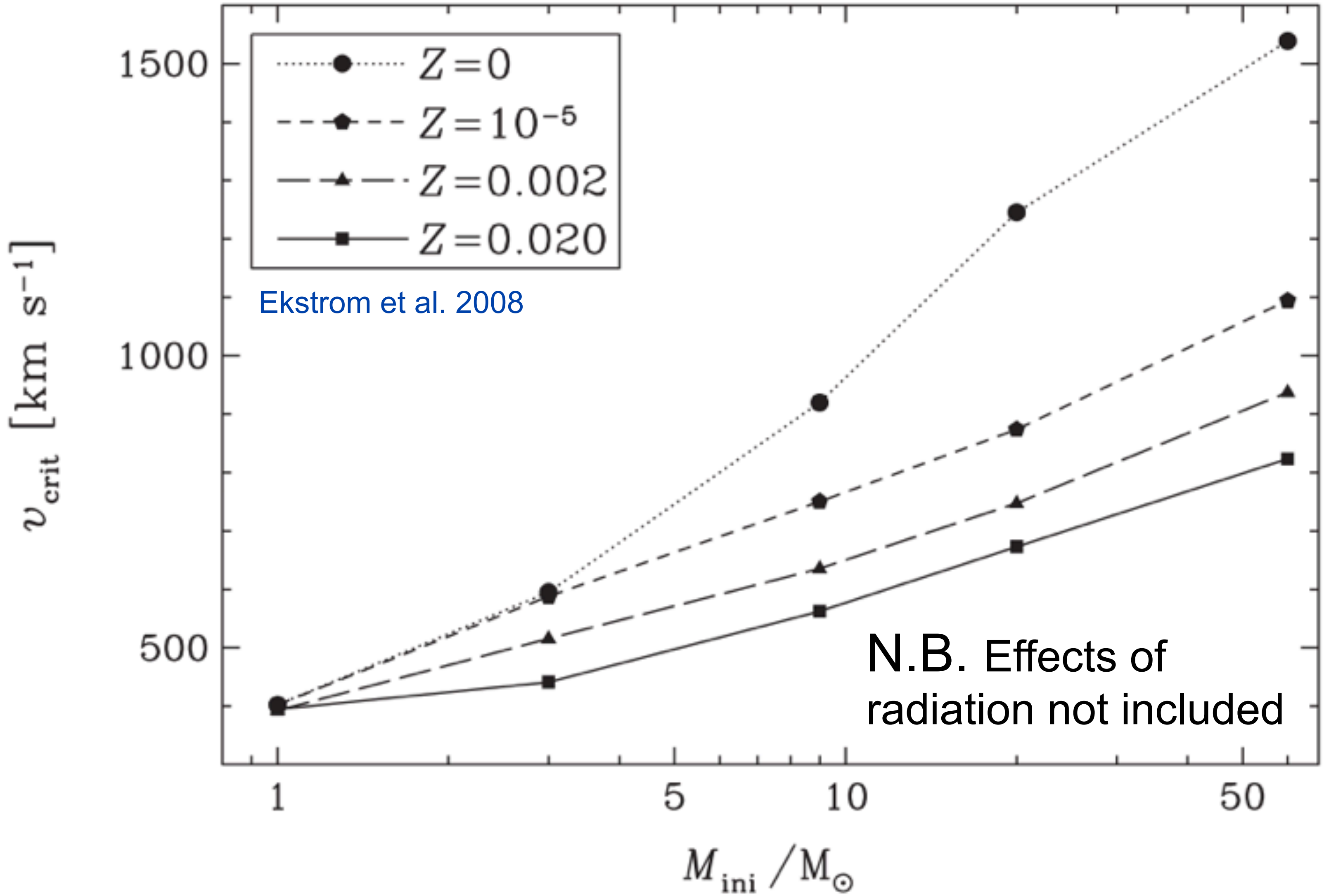


Baroclinicity leads to instabilities



Mass-loss

Critical Velocity



Critical Velocity

Gamma-Omega Limit

$$\vec{g}_{\mathrm{tot}} = \vec{g}_{\mathrm{grav}} + \vec{g}_{\mathrm{cen}} + \vec{g}_{\mathrm{rad}}$$

$$\boxed{v^2_{\mathrm{crit}} \equiv \frac{Gm}{r}(1 - \Gamma)} \qquad \Gamma \equiv \frac{\kappa L}{4\pi c G m}$$

Rotationally enhanced Mass-loss

Theoretically one expects additional mass-loss as the star approaches critical rotation, as material becomes less and less bound to the stellar surface.

$$\vec{g}_{\rm tot} = \vec{g}_{\rm grav} + \vec{g}_{\rm cen} + \vec{g}_{\rm rad} \qquad \Gamma \equiv \frac{\kappa L}{4\pi c G m}$$

$$v^2_{\rm crit} \equiv \frac{Gm}{r}(1-\Gamma) \qquad \Omega \equiv \frac{v}{v_{\rm crit}}$$

$$\dot{M}(\omega) \equiv \dot{M}(\omega = 0) \times \left(\frac{1}{1-\Omega}\right)^{\xi}, \quad \xi \approx 0.43\ ,$$

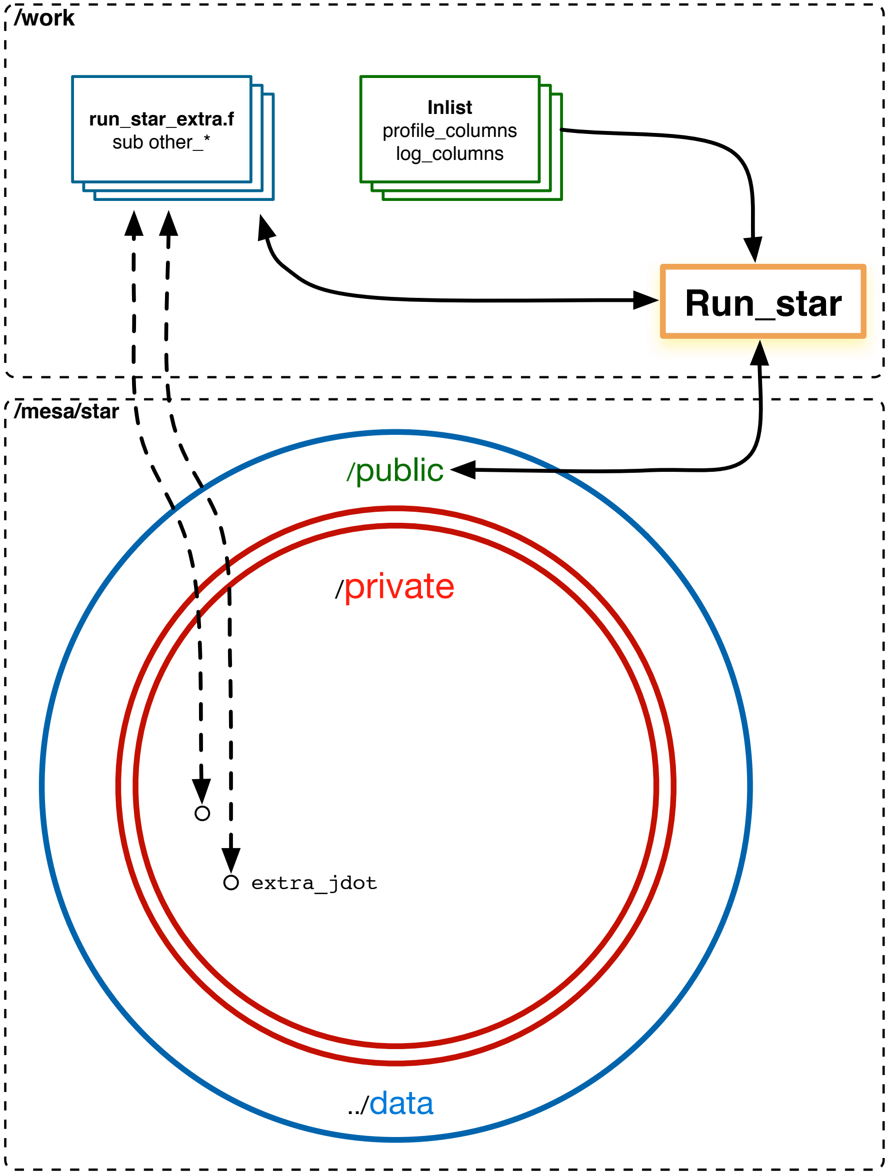
Langer, 1998

See Georgy et al. 2011 for a more thorough discussion of rotationally enhanced mass-loss

MESA Star

- /private/winds.f

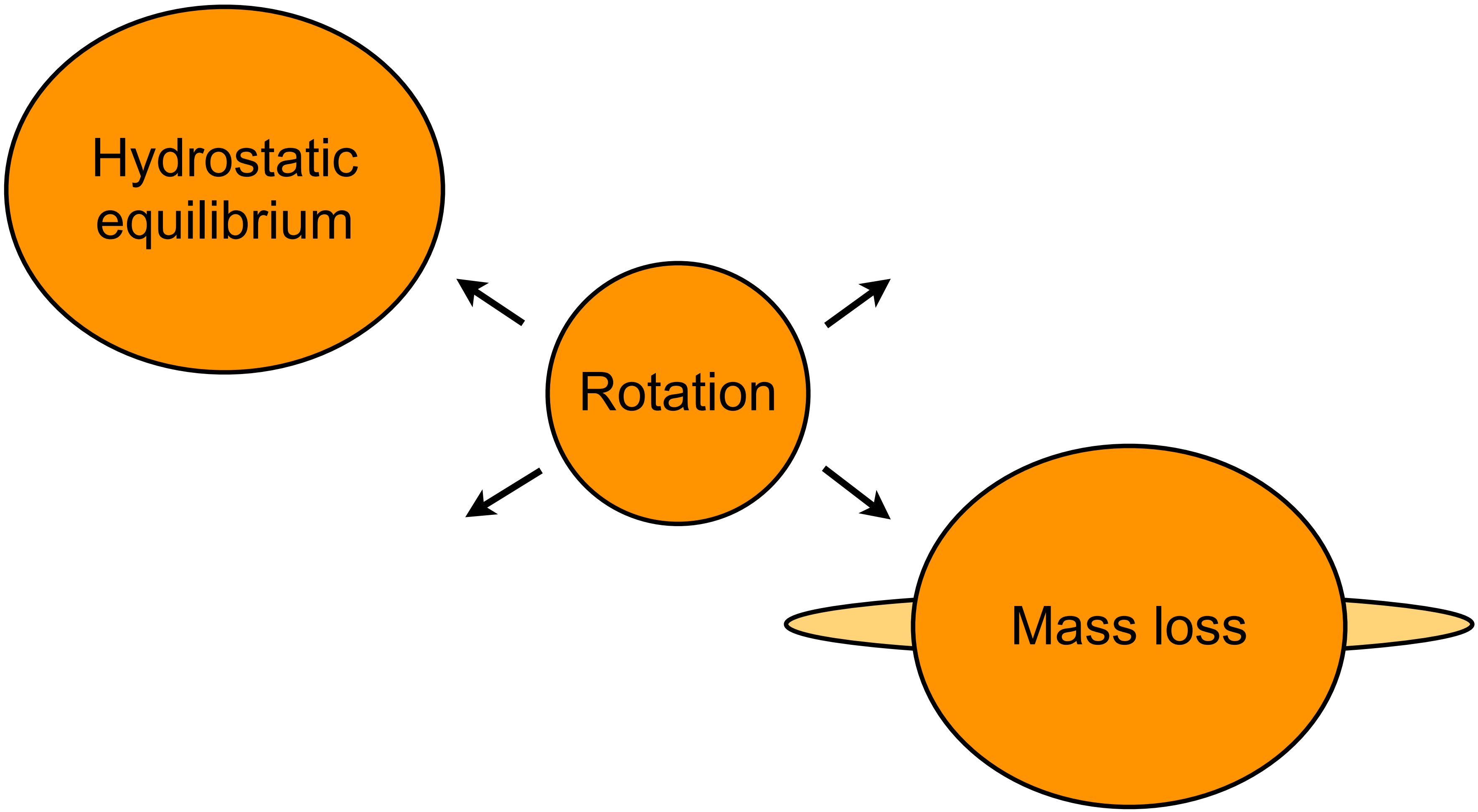
```
subroutine rotation_enhancement(ierr)

   integer, intent(out) :: ierr
   ! as in Heger, Langer, and Woosley, 2000, ApJ, 528:368-396.  section 2.6
   ! Mdot = Mdot_no_rotation/(1 - Osurf/Osurf_crit)^mdot_omega_power
   ! where Osurf = angular velocity at surface
   !       Osurf_crit^2 = (1 - Gamma_edd)*G*M/R_equitorial^3
   !       Gamma_edd = kappa*L/(4 pi c G M), Eddington factor
   real(dp) :: enhancement, wind_mdot, &
      kh_timescale, mdot_lim, mdot_prev, &
      dmsfac, dmskhf, wind_mdot_lim

   include 'formats.dek'

   ierr = 0
   if (.not. s% rotation_flag) return
   if (s% mdot_omega_power <= 0) return
   if (s% mstar_dot >= 0) return

   wind_mdot = -s% mstar_dot

   kh_timescale = s% cgrav(1)*M1**2/(2*R1*L1)
   dmskhf = s% rotational_mdot_kh_fac
   dmsfac = s% rotational_mdot_boost_fac
   wind_mdot_lim = min(dmskhf*M1/kh_timescale, wind_mdot*dmsfac)

   enhancement = max(1d-22, 1d0 - s% w_div_w_crit_avg_surf)**(-s% mdot_omega_power)
   if (enhancement > s% lim_trace_rotational_mdot_boost) then
      write(*,1) 'mdot rotation_enhancement factor', &
         enhancement, s% w_div_w_crit_avg_surf
   end if
   if (s% max_rotational_mdot_boost > 0 .and. &
         enhancement > s% max_rotational_mdot_boost) then
      enhancement = s% max_rotational_mdot_boost
      !write(*,1) 'reduce to max_rotational_mdot_boost', &
      !      s% max_rotational_mdot_boost
```



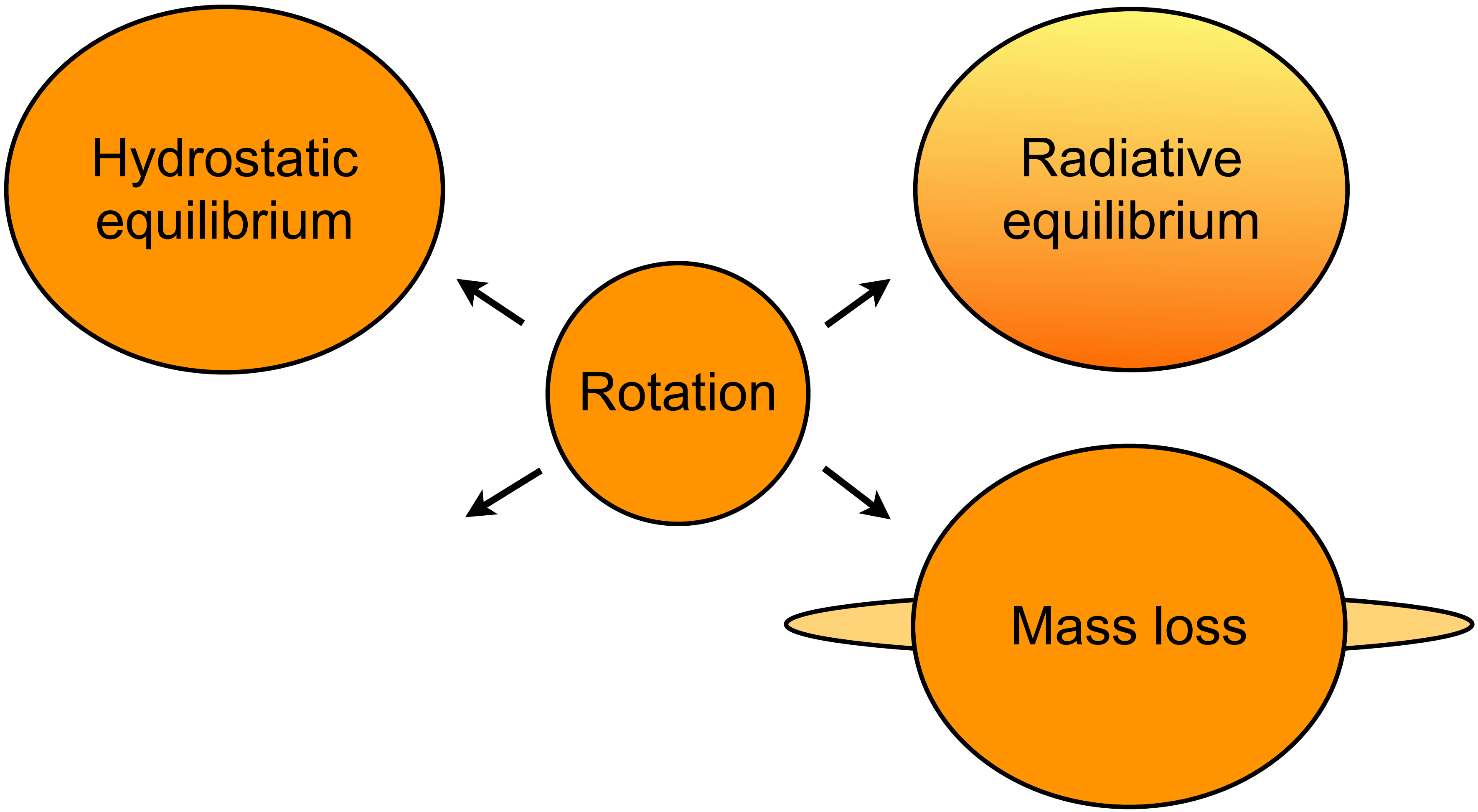
/private/winds.f

```fortran
subroutine rotation_enhancement(ierr)

   integer, intent(out) :: ierr
   ! as in Heger, Langer, and Woosley, 2000, ApJ, 528:368-396.  section 2.6
   ! Mdot = Mdot_no_rotation/(1 - Osurf/Osurf_crit)^mdot_omega_power
   ! where Osurf = angular velocity at surface
   !       Osurf_crit^2 = (1 - Gamma_edd)*G*M/R_equitorial^3
   !       Gamma_edd = kappa*L/(4 pi c G M), Eddington factor
   real(dp) :: enhancement, wind_mdot, &
      kh_timescale, mdot_lim, mdot_prev, &
      dmsfac, dmskhf, wind_mdot_lim

   include 'formats.dek'

   ierr = 0
   if (.not. s% rotation_flag) return
   if (s% mdot_omega_power <= 0) return
   if (s% mstar_dot >= 0) return

   wind_mdot = -s% mstar_dot

   kh_timescale = s% cgrav(1)*M1**2/(2*R1*L1)
   dmskhf = s% rotational_mdot_kh_fac
   dmsfac = s% rotational_mdot_boost_fac
   wind_mdot_lim = min(dmskhf*M1/kh_timescale, wind_mdot*dmsfac)

   enhancement = max(1d-22, 1d0 - s% w_div_w_crit_avg_surf)**(-s% mdot_omega_power)
   if (enhancement > s% lim_trace_rotational_mdot_boost) then
      write(*,1) 'mdot rotation_enhancement factor', &
         enhancement, s% w_div_w_crit_avg_surf
   end if
   if (s% max_rotational_mdot_boost > 0 .and. &
         enhancement > s% max_rotational_mdot_boost) then
      enhancement = s% max_rotational_mdot_boost
      !write(*,1) 'reduce to max_rotational_mdot_boost', &
      !      s% max_rotational_mdot_boost
```

Effects of rotation



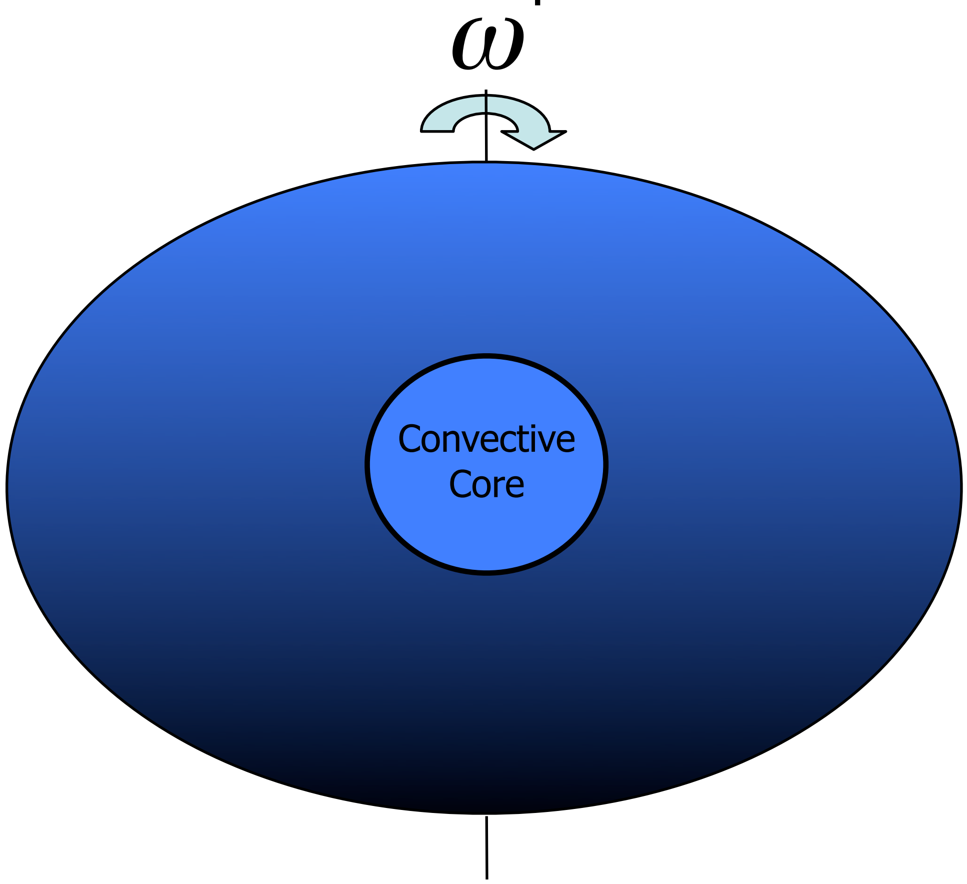
Effects of rotation



Radiative Equilibrium

The Von Zeipel Theorem

The thermal flux through the surface of a (radiative) rotating star is proportional to the local effective gravity. Since this depends on the co-latitude, one expects a greater radiative flux at the poles than at the equator.

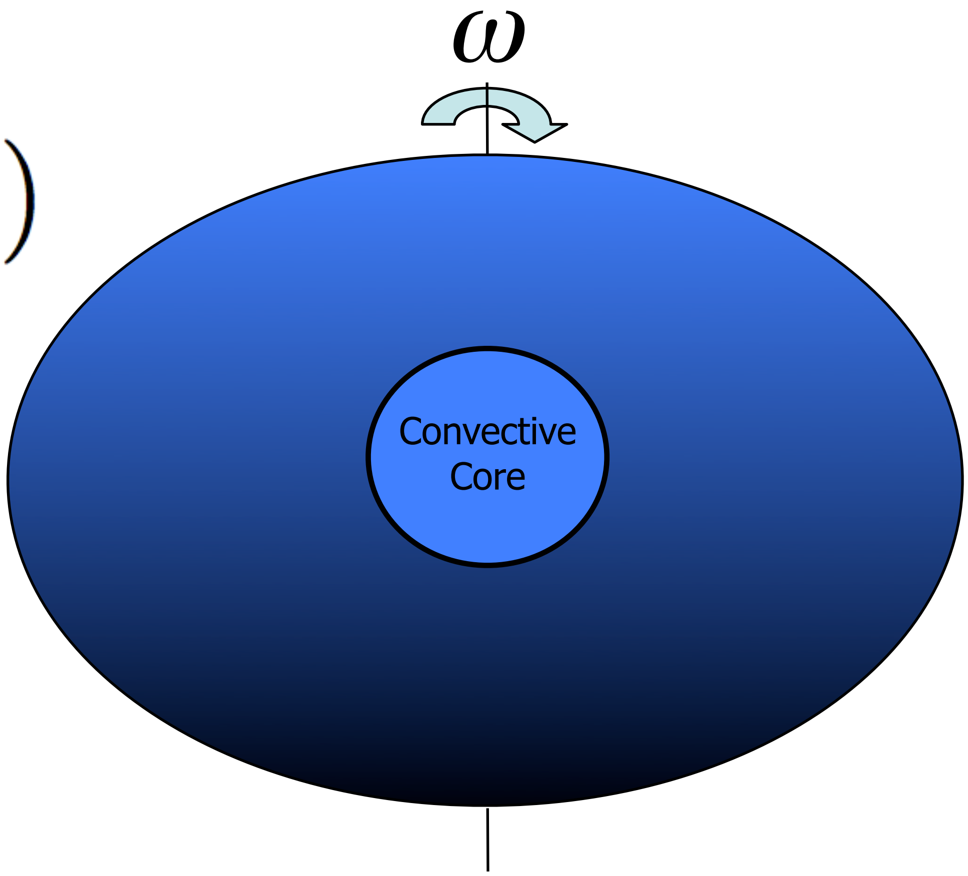


Von Zeipel (1924)

The Von Zeipel Theorem

The thermal flux through the surface of a (radiative) rotating star is proportional to the local effective gravity. Since this depends on the co-latitude, one expects a greater radiative flux at the poles than at the equator.

$$T_{\mathrm{eff}}(\vartheta) \sim g_{\mathrm{eff}}^{1/4}(\vartheta)$$



Von Zeipel (1924)

The Von Zeipel Theorem

In the case of solid body rotation:

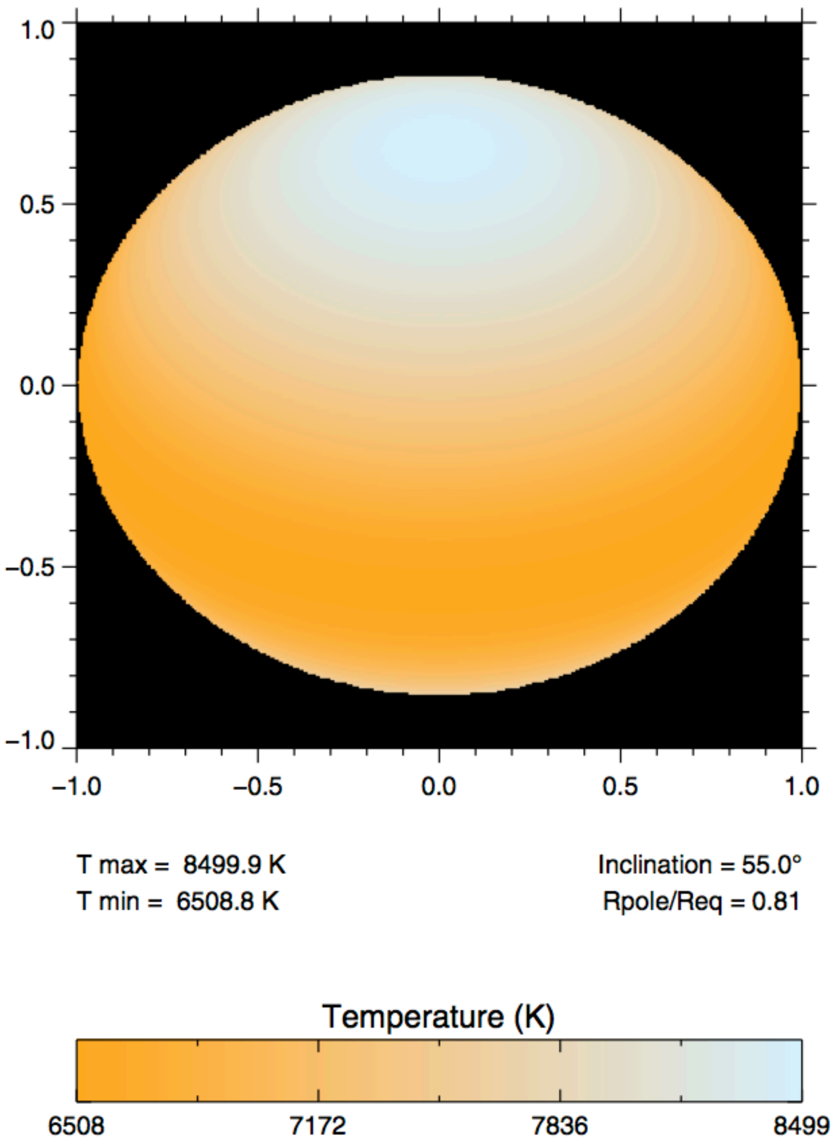
Von Zeipel (1924)

$$\vec{F}(\Omega, \vartheta) = -\frac{L}{4\pi G M^*}\vec{g}_{\rm eff}(\Omega, \vartheta)$$

$$T_{\rm eff}(\Omega, \vartheta) = \left(\frac{L}{4\pi\sigma G M^*}\right)^{1/4}[g_{\rm eff}(\Omega, \vartheta)]^{1/4}$$

Both g_{eff} and T_{eff} vary over the surface of a rotating star and influence the emergent spectrum. The equatorial regions are fainter and cooler than the polar ones, which are brighter and hotter (**the differences of T_{eff} may reach a factor of 2**). This effect is called the gravity darkening.The von Zeipel theorem in a star with shellular rotation shows only minor differences
(Maeder, 1999, Maeder & Meynet 2012)

Interferometry of rotating stars

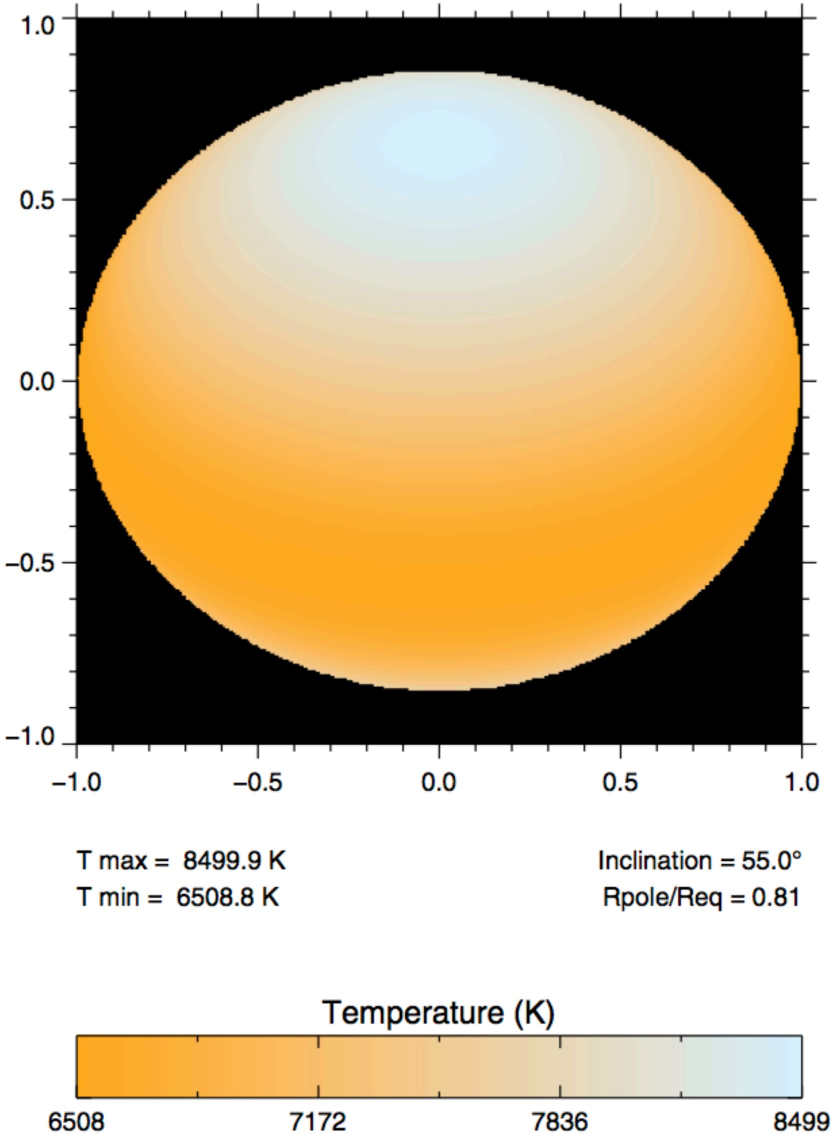


Altair (A7IV-V Star)
VLTI Observations

Von Zeipel 'gravity darkening' Confirmed

Domiciano Da Souza et al.2005

Interferometry of rotating stars



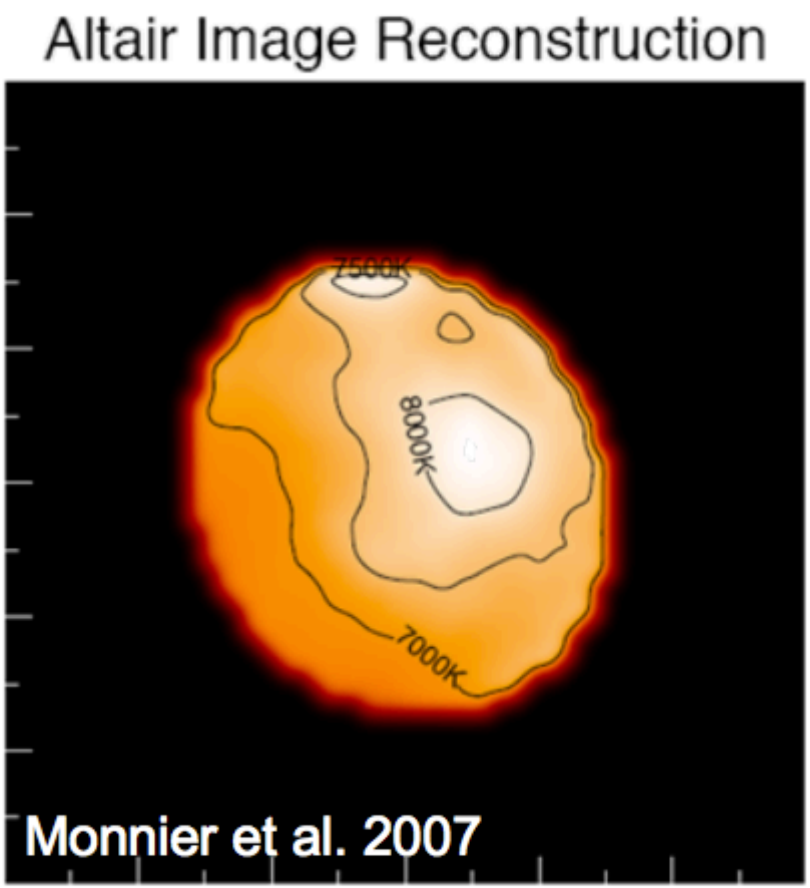
Altair (A7IV-V Star)
VLTI Observations

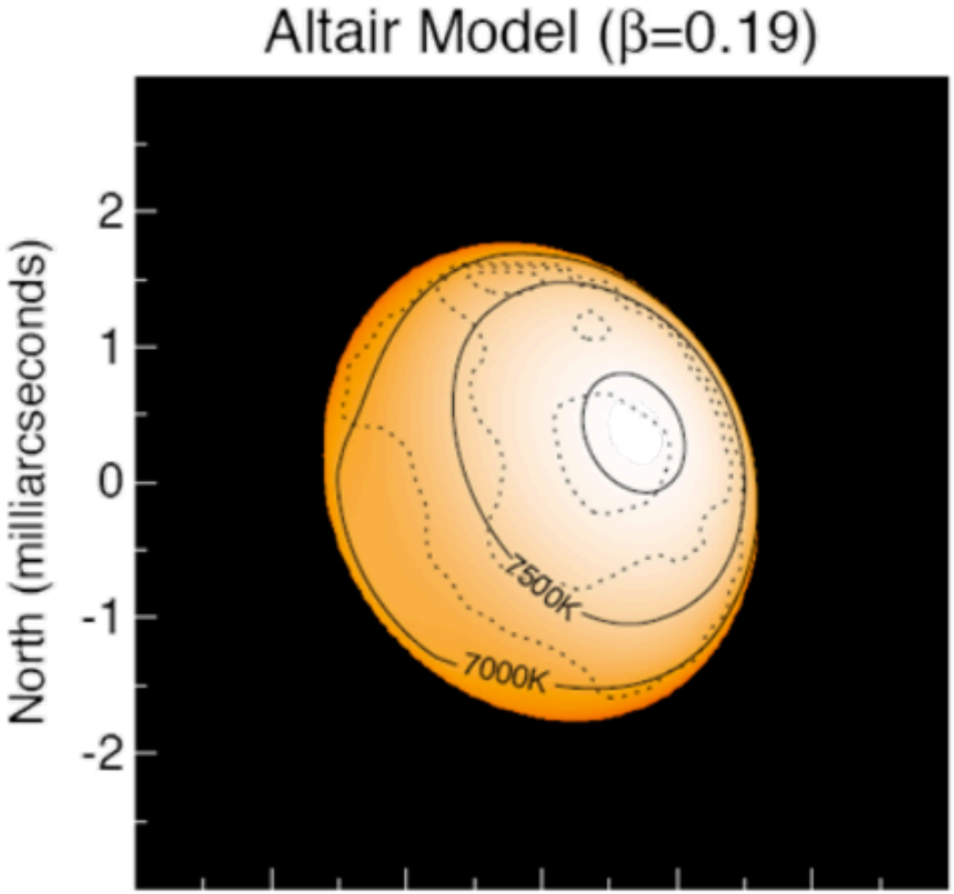
Von Zeipel 'gravity darkening' Confirmed

$$T_{\text{eff}}(\vartheta) \sim g_{\text{eff}}^{1/4}(\vartheta)$$

Domiciano Da Souza et al.2005

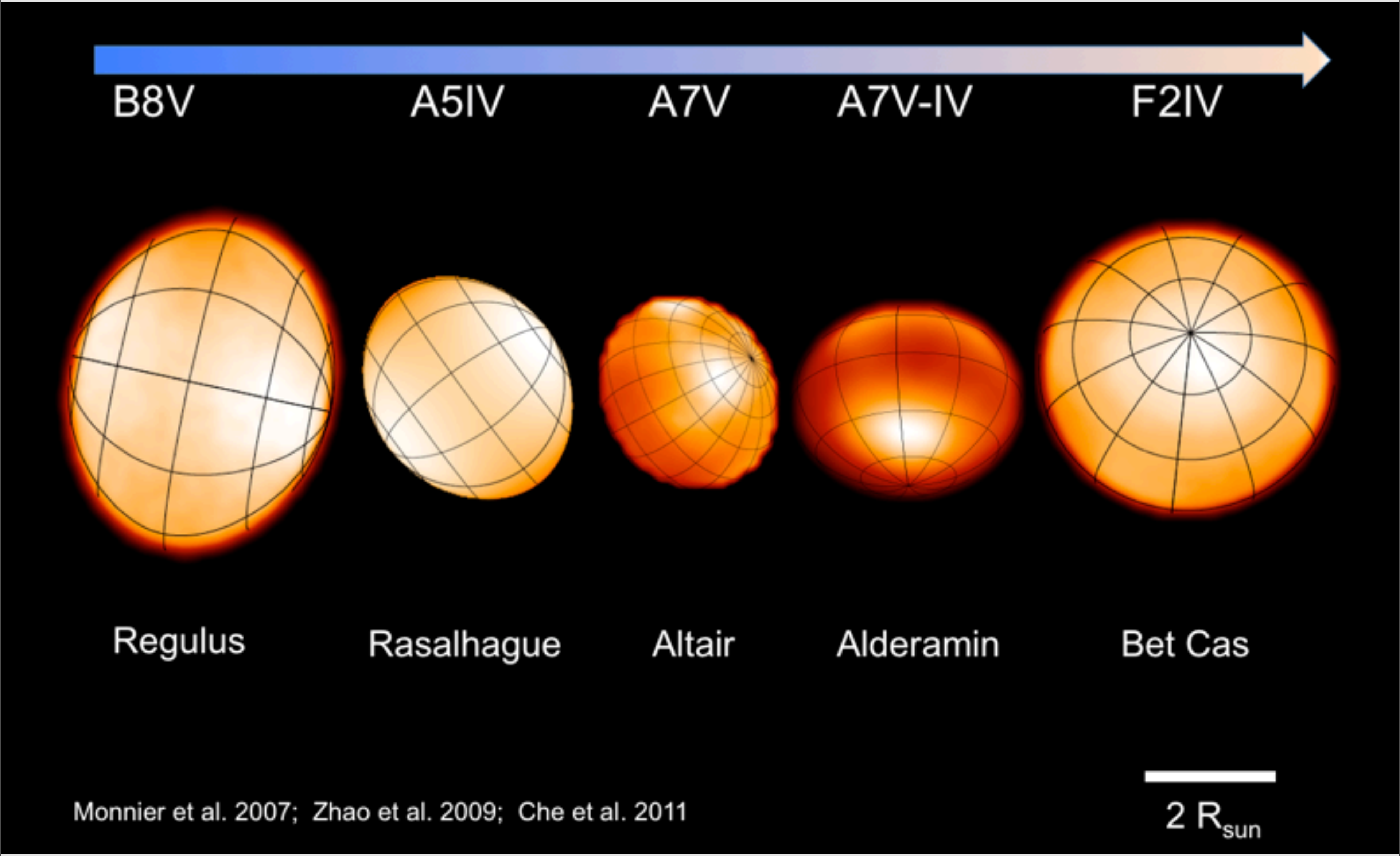
Interferometry of rotating stars





Altair (A7IV-V Star): Von Zeipel 'gravity darkening' but with a different exponent

Interferometry of rotating stars



Modeling Rotating Stars

The computational Challenge

- Stars are rotating balls of hot plasma
- They undergo a variety of instabilities
- Large range of timescales
- It would be ideal to perform fully (magneto) hydrodynamic calculations. But this is currently computationally impossible.
- At this time 3D stellar evolution calculations can evolve a few hours of a star life.
- How can we model rotation in a 1D, implicit code?

The Shellular Approximation

- Rotation and especially differential rotation generates turbulent motions. On the Earth, we have the example of west winds and jet streams. In a radiative zone, the turbulence is stronger (Zahn, 1992) in the horizontal than in the vertical direction, because in the vertical direction the stable thermal gradient opposes a strong force to the fluid motions
- According to Zahn (1975), Chaboyer & Zahn (1992), and Zahn (1992), anisotropic turbulence acts much stronger on isobars than in the perpendicular direction. This enforces a shellular rotation law (Meynet & Maeder 1997), and it sweeps out compositional differences on isobars. Therefore it can be assumed that matter on isobars is approximately chemically homogeneous. Together with the shellular rotation, this allows us to retain a **one-dimensional approximation**. The specific angular momentum, j, of a mass shell is treated as a local variable, and the angular velocity, omega, is computed from the specific moment of inertia, i. (Heger et al. 2000)
- In this approach, mass shells correspond to isobars instead of spherical shells.

The structure equations of rotating stars

For a star in **shellular** rotation it is possible to modify the eqs of stellar structure to include the effect of the centrifugal force while keeping the form of the equations very close to that of the non-rotating case. Basically all quantities are redefined on isobars.

Mass conservation

$$\boxed{\frac{\mathrm{d}m_P}{\mathrm{d}r_P} = 4\pi r_P^2 \rho}$$

Hydrostatic Eq.

$$\boxed{\frac{\mathrm{d}P}{\mathrm{d}m_P} = -\frac{Gm_P}{4\pi r_P^4} f_P}$$

Energy transport

$$\boxed{\frac{\mathrm{d}\ln T}{\mathrm{d}\ln P} = \frac{3\kappa P L_P}{16\pi acGm_P T^4}\frac{f_T}{f_P}}$$

...

$$V_P = 4\pi r_P^3/3$$

$$\langle q \rangle \equiv \frac{1}{S_P}\oint_{S_P} q\,\mathrm{d}\sigma,$$

$$f_P = \frac{4\pi r_P^4}{Gm_P S_P}\left\langle g_{\mathrm{eff}}^{-1}\right\rangle^{-1}$$

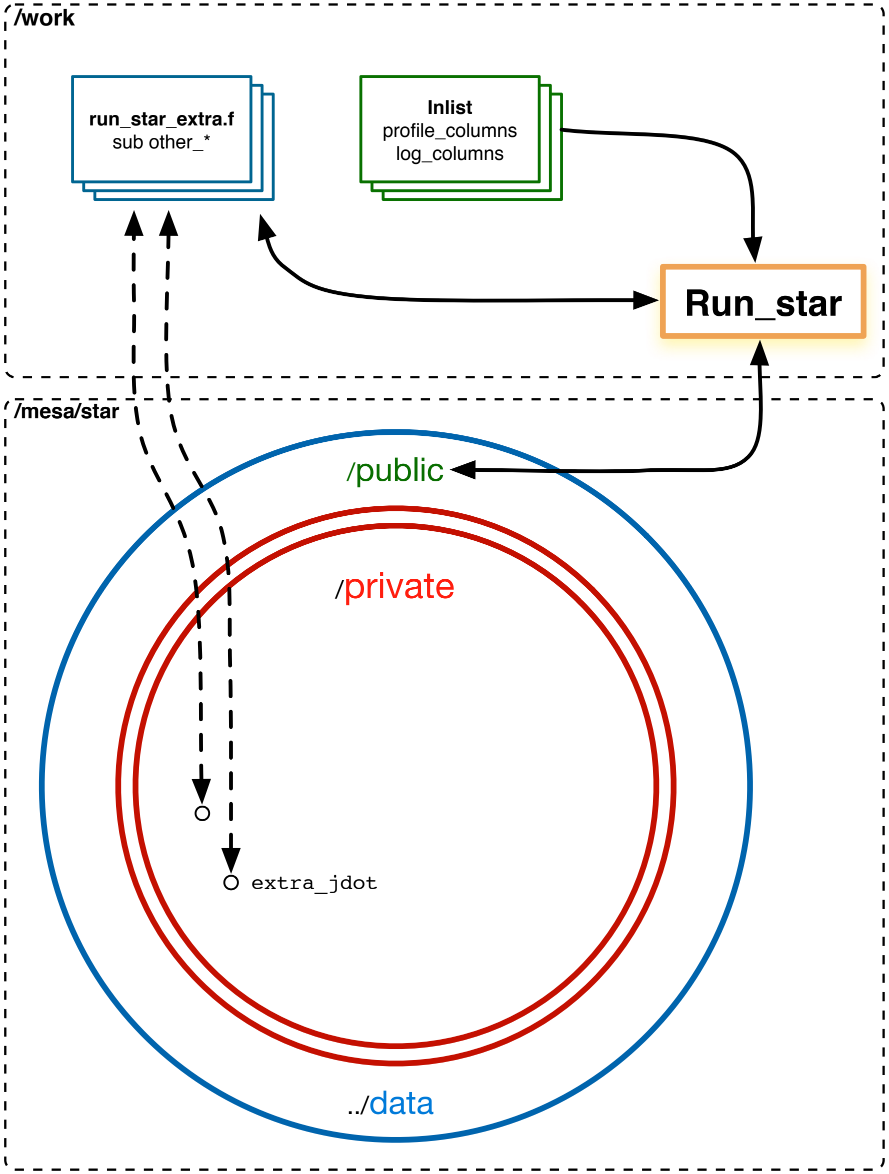
$$f_T \equiv \left(\frac{4\pi r_P^2}{S_P}\right)\left(\langle g_{\mathrm{eff}}\rangle\langle g_{\mathrm{eff}}^{-1}\rangle\right)^{-1}$$

Endal & Sofia 1978

MESA Star

- /private/hydro_rotation.f

```
! Input variables:
!  N     Number of meshpoints used by the model (arrays are this size)
!  XM    Mass coordinate [gram]
!  R     Radius coordinate [cm]
!  RHO   Density [gram/cm^3]
!  AW    Angular velocity [rad/sec]
! Output variables:
!  Correction factor FT at each meshpoint
!  Correction factor FP at each meshpoint
!  r_polar, r_equitorial at each meshpoint
subroutine eval_fp_ft( &
      nz, xm, r, rho, aw, ft, fp, r_polar, r_equitorial, report_ierr, ierr)
   use num_lib
   integer, intent(in) :: nz
   real(dp), intent(in) :: aw(:), r(:), rho(:), xm(:) ! (nz)
   real(dp), intent(out) :: ft(:), fp(:), r_polar(:), r_equitorial(:) ! (nz)
   logical, intent(in) :: report_ierr
   integer, intent(out) :: ierr
```



/private/hydro_rotation.f

```
! Input variables:
!  N     Number of meshpoints used by the model (arrays are this size)
!  XM    Mass coordinate [gram]
!  R     Radius coordinate [cm]
!  RHO   Density [gram/cm^3]
!  AW    Angular velocity [rad/sec]
! Output variables:
!  Correction factor FT at each meshpoint
!  Correction factor FP at each meshpoint
!  r_polar, r_equitorial at each meshpoint
subroutine eval_fp_ft( &
      nz, xm, r, rho, aw, ft, fp, r_polar, r_equitorial, report_ierr, ierr)
   use num_lib
   integer, intent(in) :: nz
   real(dp), intent(in) :: aw(:), r(:), rho(:), xm(:) ! (nz)
   real(dp), intent(out) :: ft(:), fp(:), r_polar(:), r_equitorial(:) ! (nz)
   logical, intent(in) :: report_ierr
   integer, intent(out) :: ierr
```

Diffusion Equations

Transport of chemical species:

$$\boxed{\left(\frac{\partial X_n}{\partial t}\right)_m = \left(\frac{\partial}{\partial m}\right)_t \left[(4\pi r^2 \rho)^2 D \left(\frac{\partial X_n}{\partial m}\right)_t\right] + \left(\frac{\mathrm{d}X_n}{\mathrm{d}t}\right)_{\mathrm{nuc}}}$$

$$t_{\mathrm{diff}} \simeq \frac{R^2}{D}$$

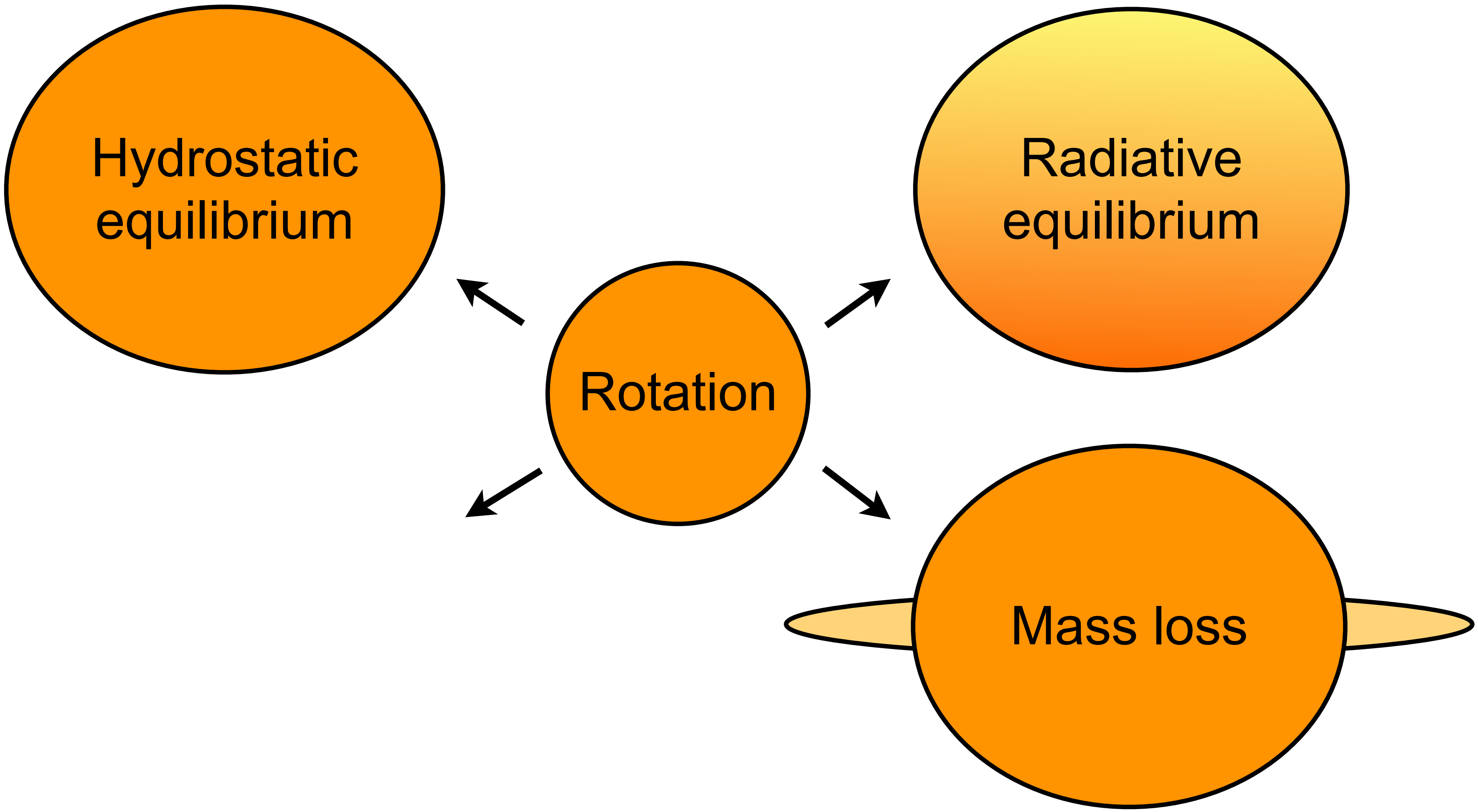
Transport of angular momentum:

$$\boxed{\left(\frac{\partial \omega}{\partial t}\right)_m = \frac{1}{i}\left(\frac{\partial}{\partial m}\right)_t \left[(4\pi r^2 \rho)^2 i v \left(\frac{\partial \omega}{\partial m}\right)_t\right] - \frac{2\omega}{r}\left(\frac{\partial r}{\partial t}\right)_m \left(\frac{1}{2}\frac{d\ln i}{d\ln r}\right)}$$

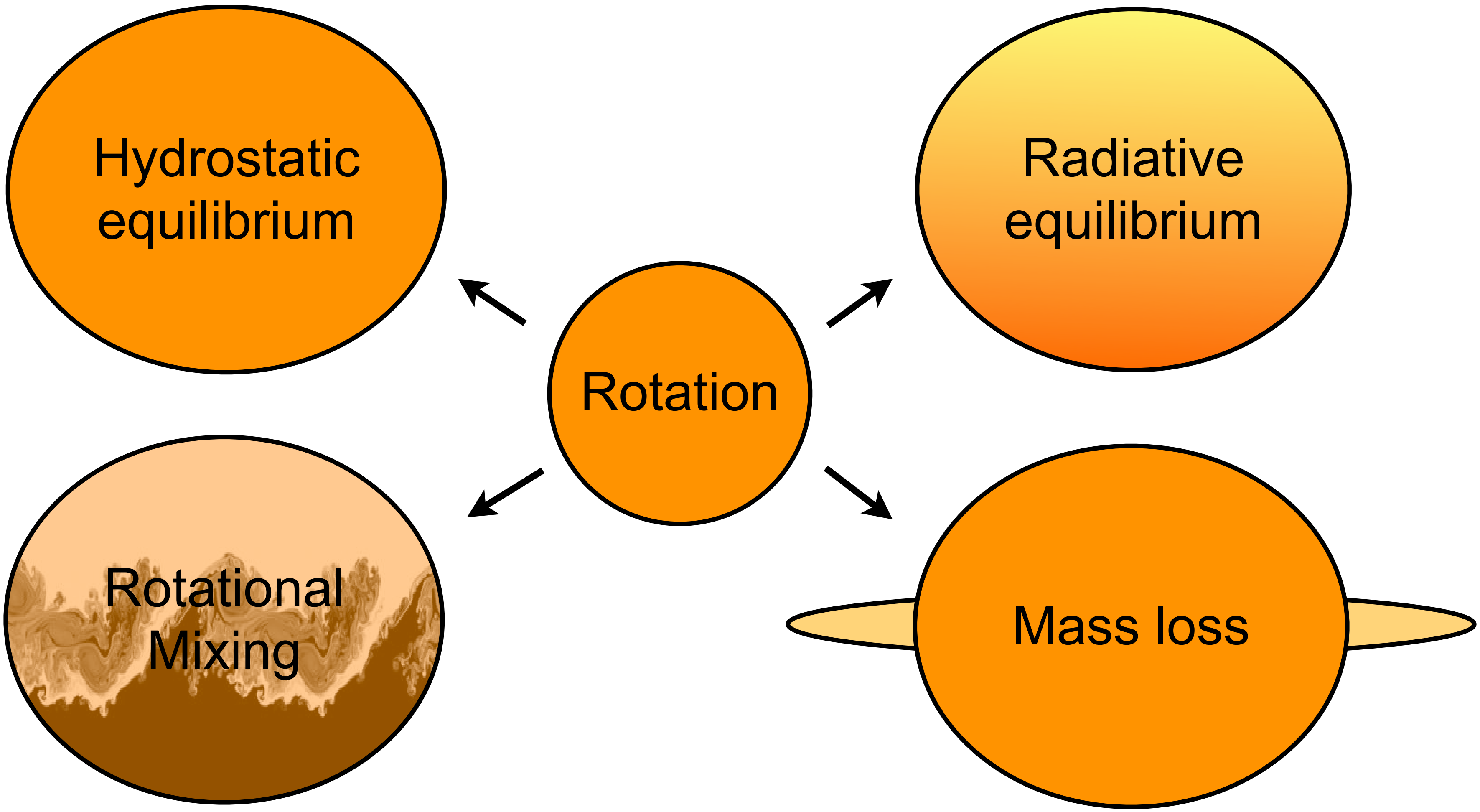
Advection- Diffusion Scheme

- An alternative approach is to solve an advection-diffusion equation for the transport of angular momentum (e.g. in the GENEVA code or the ROSE code). However, as MESA adopts the diffusion approximation (like e.g. KEPLER and STERN), I will not discuss the advection-diffusion scheme here. Nevertheless I suggest you have a look at the relevant literature (e.g. Mader & Meynet 2000, 2012)
- A very interesting work comparing the different implementations has been published by Potter et al. 2012. While the advection-diffusion scheme is mathematically more sound, a key result of Potter's work is that is not yet possible to prefer one of the two implementations over the other, given the available observational constraints.

Effects of rotation



Effects of rotation



Rotational Instabilities

The unstable interiors of rotating stars

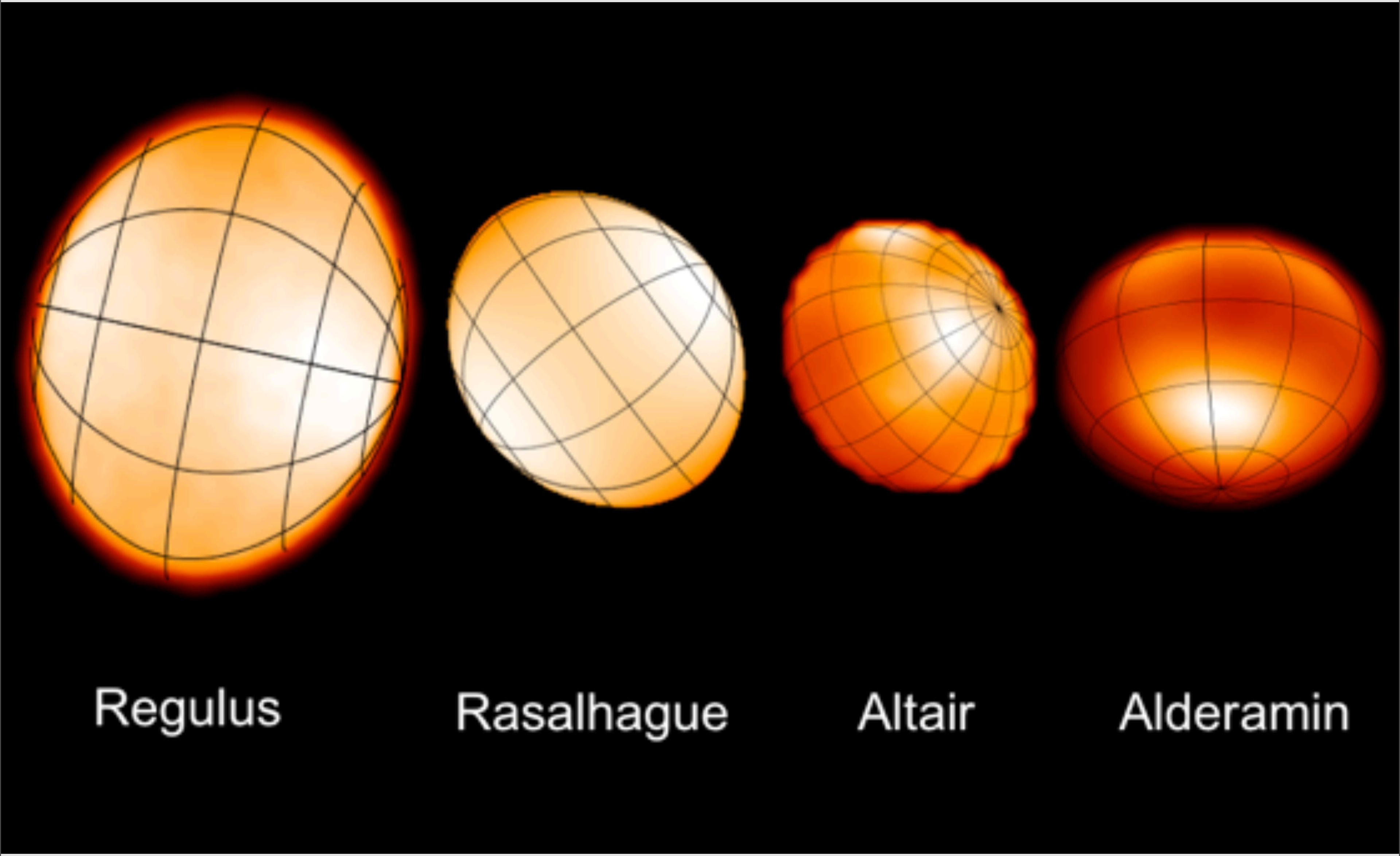
- Differential rotation is expected to arise in stars because of hydrostatic **structural evolution**, **mass loss** and **meridional circulation**. As a consequence, stars are subject to a number of local hydrodynamic instabilities.

- These instabilities arise and cause diffusion of angular momentum (and chemicals) while they try to bring the star back to solid body rotation, its lowest energy state.

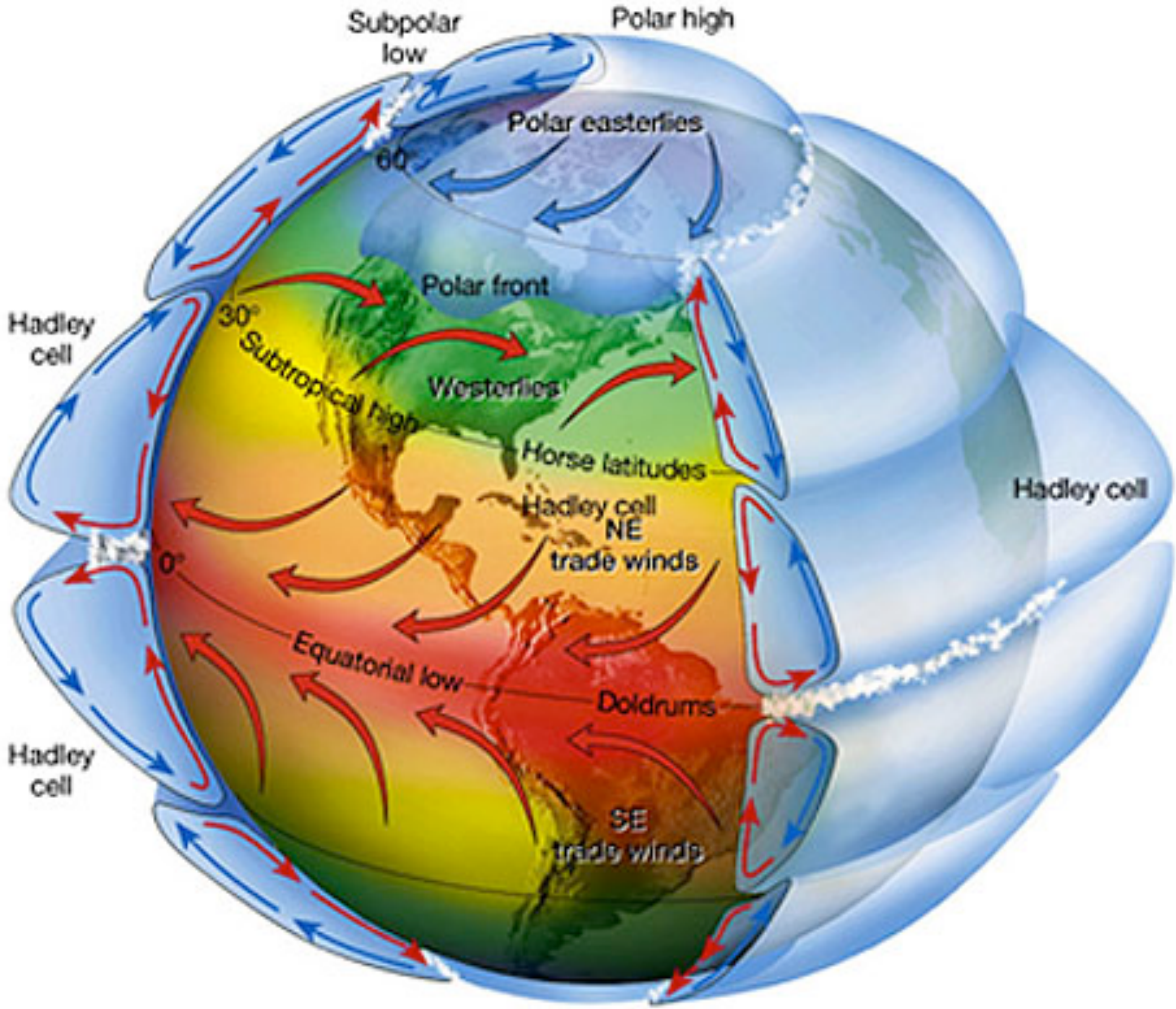
Rotational Instabilities

- Rotational “instabilities” mix rotating stars
- Different timescales: secular/dynamical instabilities
- MESA
  - (Eddington - Sweet circulation) (ES)
  - Dynamical and Secular Shear (DSI, SSI)
  - Goldreich-Schubert-Fricke (GSF)
  - Solberg - Høiland (SH)
  - (Tayler - Spruit) (TS)
- See `/mesa/star/private/rotation_mix_info.f`

Eddington - Sweet (D_ES)

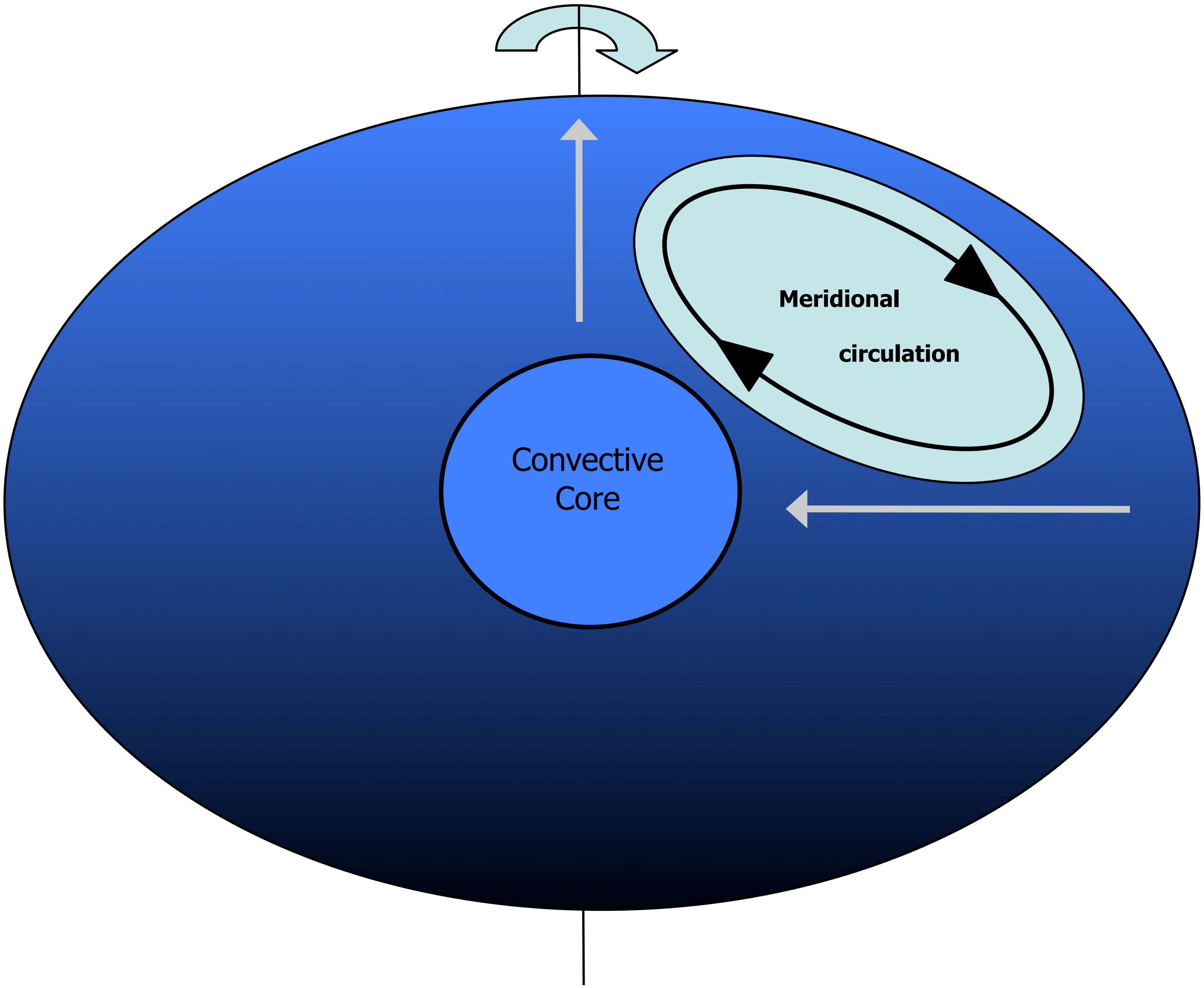


Thermal imbalance drives circulations

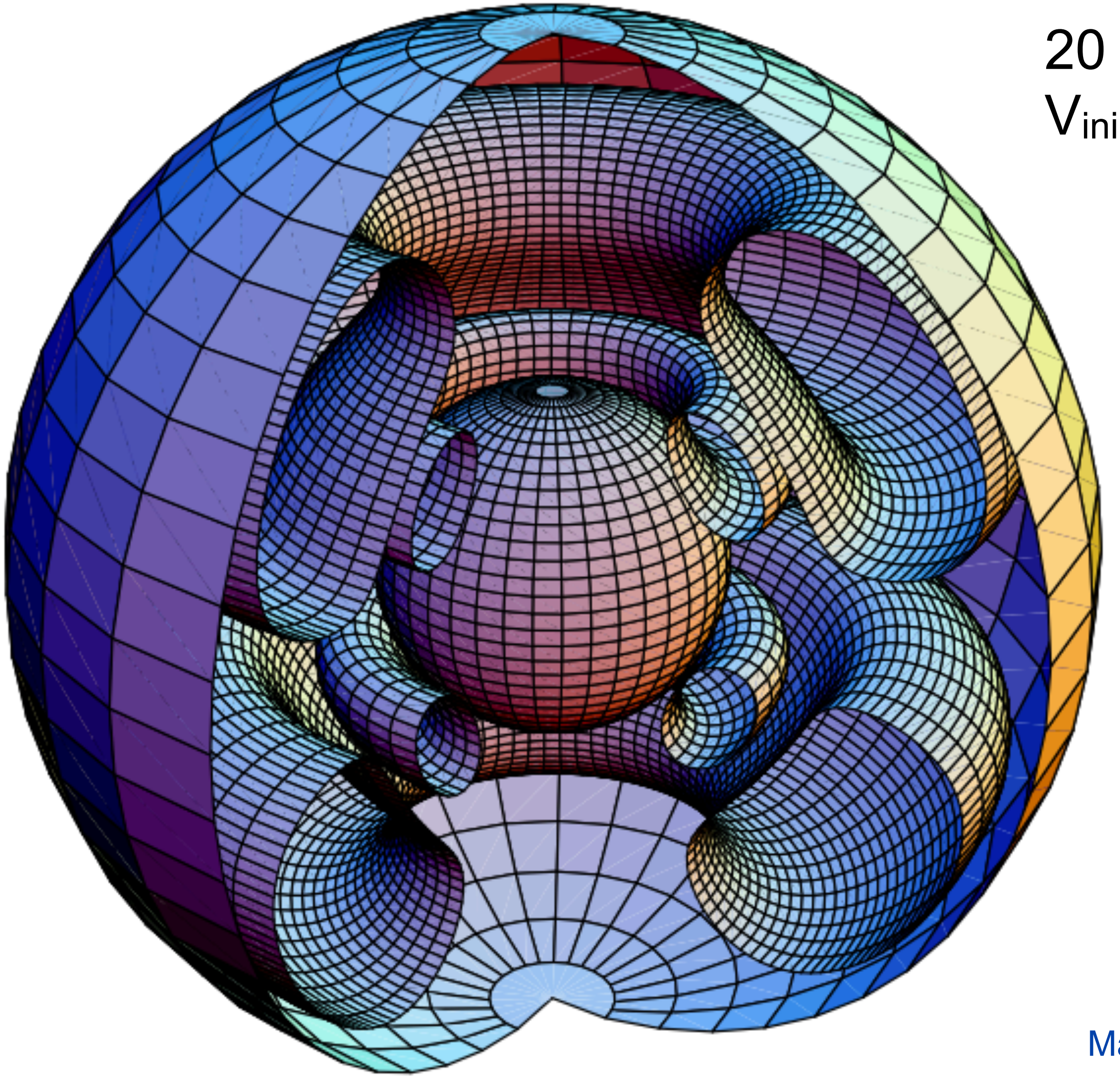


Thermal imbalance drives circulations

Thermal imbalance drives circulations



Meridional Circulation



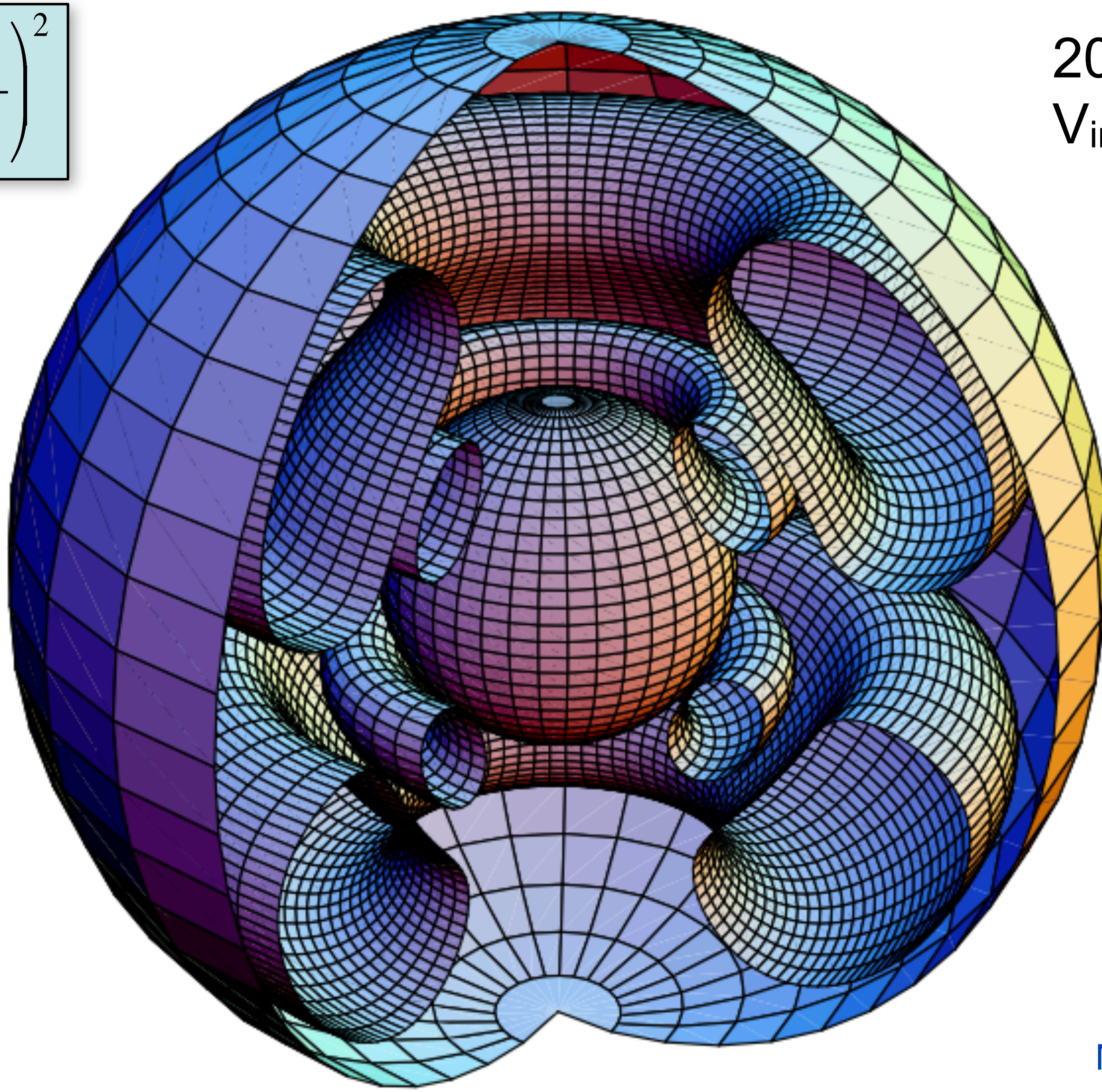
20 M_{sun}
V_{ini} = 300 km/s

Maeder & Meynet 2002

Meridional Circulation

$$\tau_{ES} \propto \tau_{KH}\left(\frac{\omega_K}{\omega}\right)^2$$

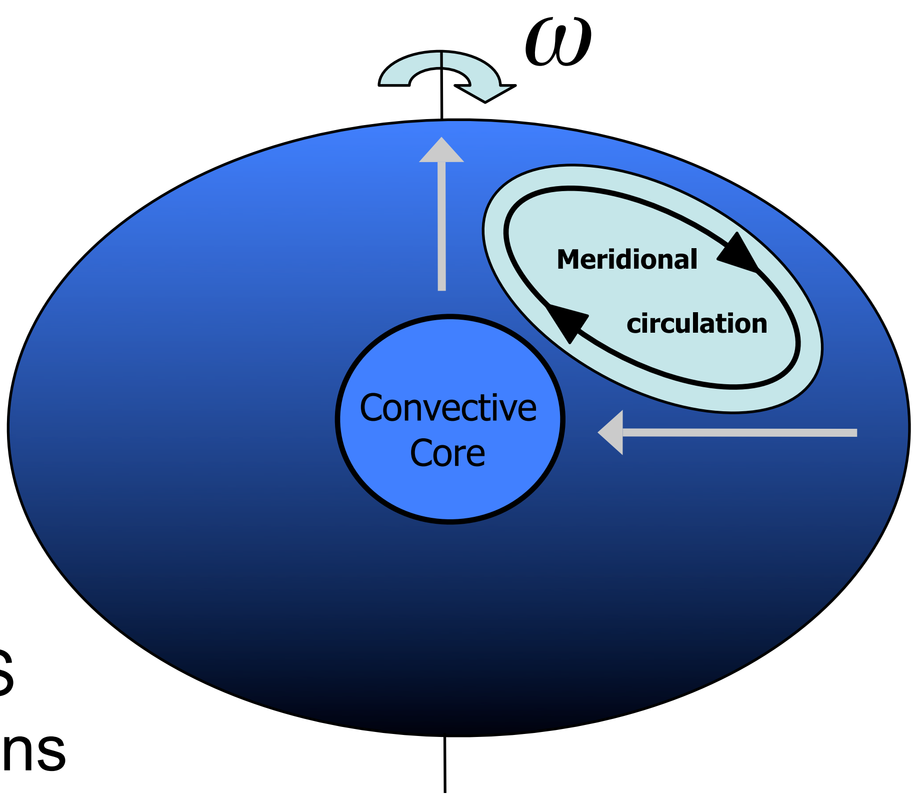
20 M_{sun}
V_{ini} = 300 km/s



Maeder & Meynet 2002

Eddington - Sweet (D_ES)

- Is a meridional circulation mixing the stellar interior
- In massive stars is one of the most efficient processes
- Mixing process on t_{KH}

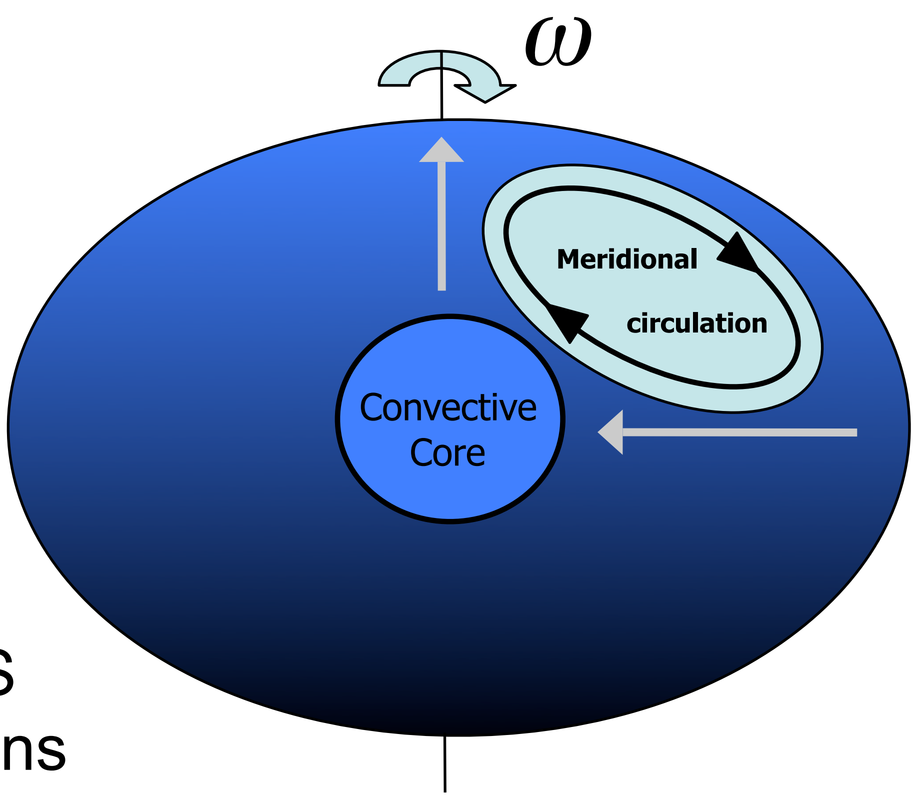


A diffusion coefficient D_ES
enters the diffusion equations

Eddington - Sweet (D_ES)

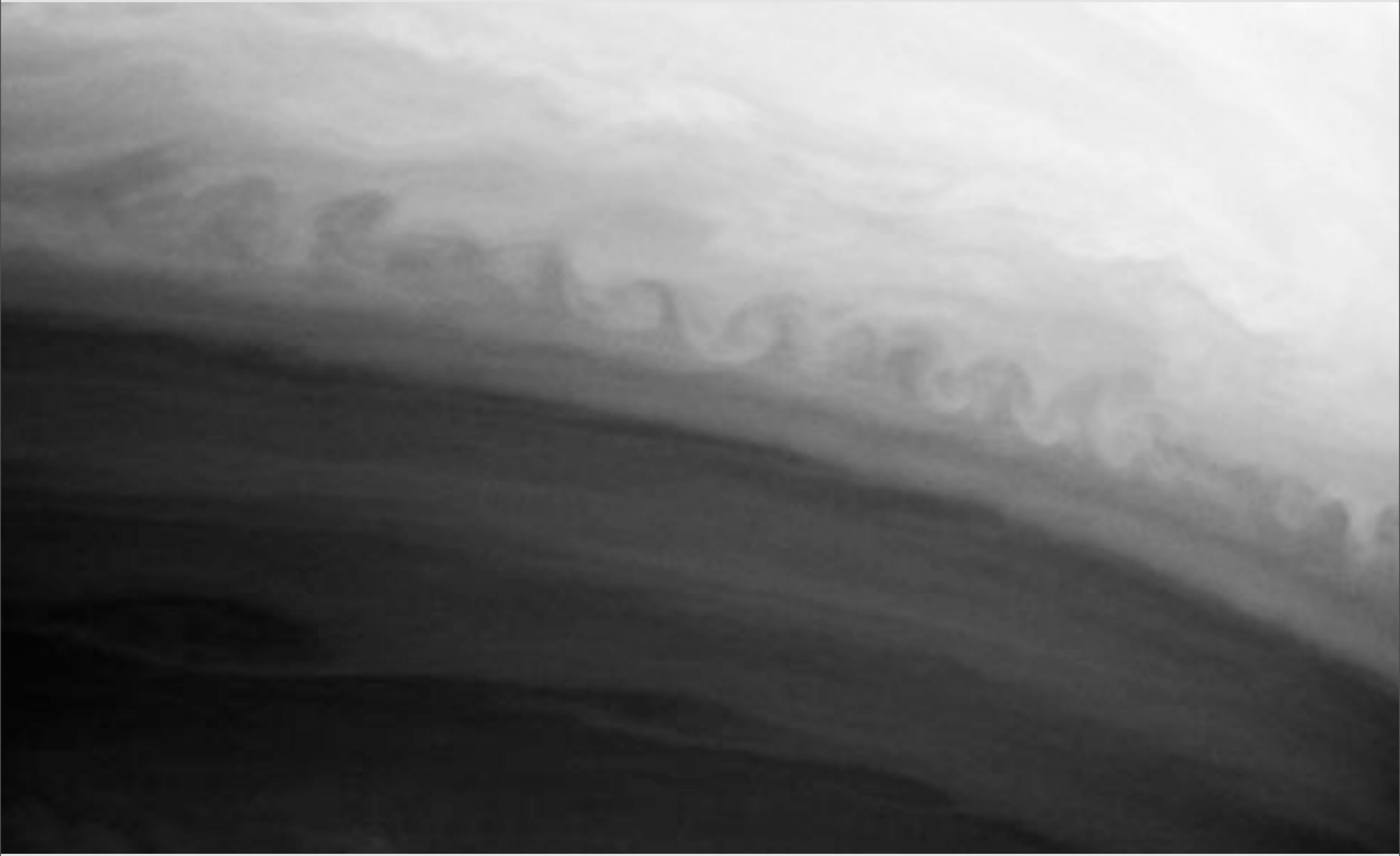
- Is a meridional circulation mixing the stellar interior
- In massive stars is one of the most efficient processes
- Mixing process on t_{KH}

$$\tau_{ES} \propto \tau_{KH} \left(\frac{\omega_K}{\omega} \right)^2$$

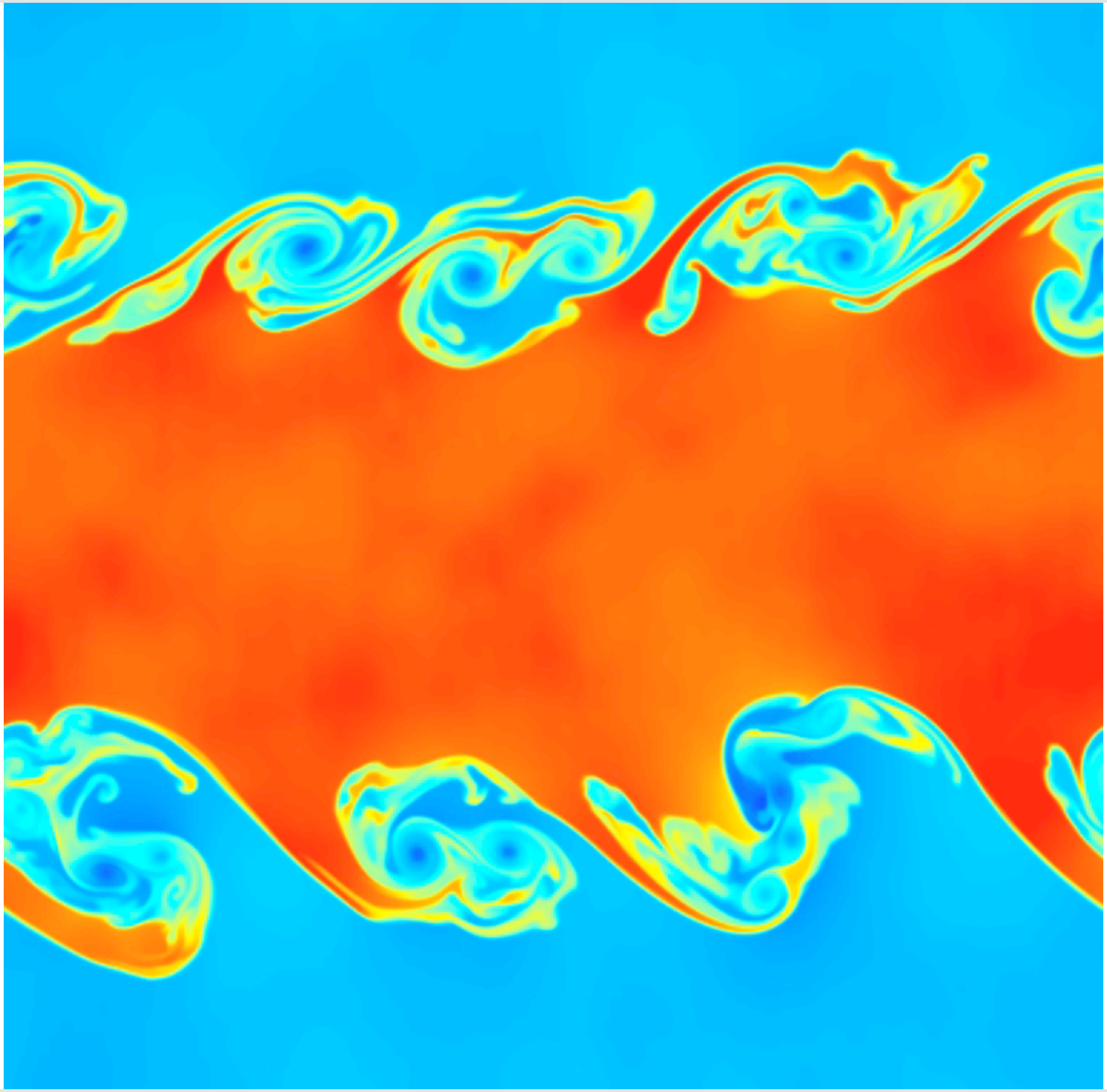


A diffusion coefficient D_ES enters the diffusion equations

Dynamical Shear (D_DSI)



Dynamical Shear (D_DSI)



Jake Simon
(JILA)

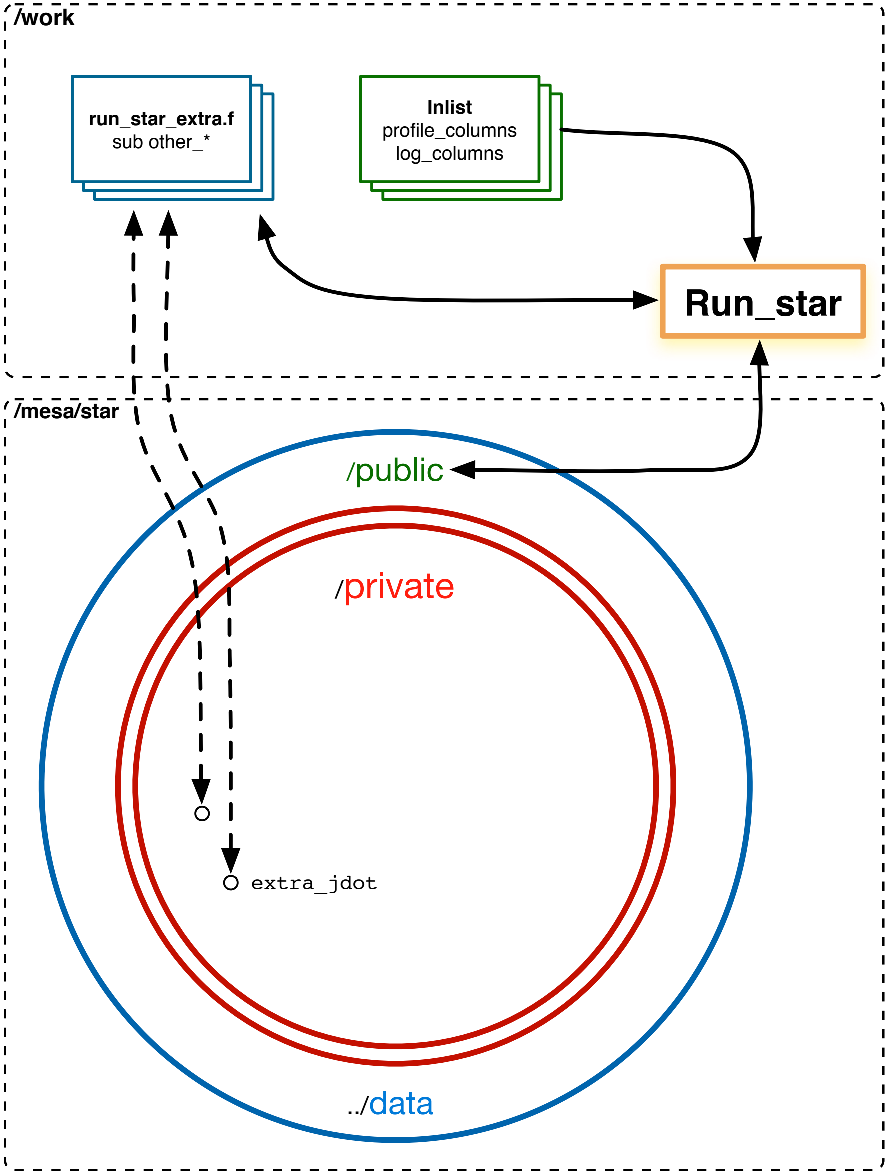
Dynamical Shear (D_DSI)

- Works very well on horizontal surfaces (isobars) and justifies (together with the baroclinic instabilities) the Shellular approximation
- In the vertical direction it is strongly inhibited by density gradients. Can only work for large degree of differential rotation
- Mixing process on $t_{Dynamical}$ $$D_{\rm shear} = \frac{1}{3} v \ell = \ldots$$
- Results in D_DSI, entering the diffusion equations

MESA Star

- /private/rotation_mix_info.f

```
subroutine set_D_DSI(ierr)
   integer, intent(out) :: ierr
   integer :: i, k, kbot, ktop
   real(dp) :: instability_height, height, D
   logical, parameter :: dbg = .false.
   include 'formats.dek'

   ierr = 0
   kbot = nz
   do i = nz-1, 1, -1
      if (Ri(i) < Ri_crit .and. s% mixing_type(i) /= convective_mixing) then
         unstable(i_DSI,i) = .true.
         if (.not. unstable(i_DSI,i+1)) kbot = i
      end if
      if (unstable(i_DSI,i+1) .and. &
            (i == 1 .or. .not. unstable(i_DSI,i)) .and. kbot > 1) then
         if (unstable(i_DSI,i)) then
            ktop = i
         else
            ktop = i+1
         end if
         instability_height = r(ktop) - r(kbot)
         if (dbg) write(*,3) 'DSI: ktop, kbot', ktop, kbot
         do k = ktop, kbot
            height = min(instability_height, scale_height(k))
            D = s% D_DSI_factor*height**2/t_dyn(k)
            s% D_DSI(k) = min(D, scale_height(k)*csound(k))
            if (dbg) write(*,2) 'D_DSI', k, s% q(k), &
               s% D_DSI(k), D, scale_height(k)*csound(k), &
               Ri(k), Ri_crit, height, t_dyn(k), &
               instability_height, scale_height(k), csound(k)
         end do
         if (dbg) write(*,*)
      end if
   end do
   if (dbg) stop 'set_D_DSI'
end subroutine set_D_DSI
```



```fortran
subroutine set_D_DSI(ierr)
   integer, intent(out) :: ierr
   integer :: i, k, kbot, ktop
   real(dp) :: instability_height, height, D
   logical, parameter :: dbg = .false.
   include 'formats.dek'

   ierr = 0
   kbot = nz
   do i = nz-1, 1, -1
      if (Ri(i) < Ri_crit .and. s% mixing_type(i) /= convective_mixing) then
         unstable(i_DSI,i) = .true.
         if (.not. unstable(i_DSI,i+1)) kbot = i
      end if
      if (unstable(i_DSI,i+1) .and. &
            (i == 1 .or. .not. unstable(i_DSI,i)) .and. kbot > 1) then
         if (unstable(i_DSI,i)) then
            ktop = i
         else
            ktop = i+1
         end if
         instability_height = r(ktop) - r(kbot)
         if (dbg) write(*,3) 'DSI: ktop, kbot', ktop, kbot
         do k = ktop, kbot
            height = min(instability_height, scale_height(k))
            D = s% D_DSI_factor*height**2/t_dyn(k)
            s% D_DSI(k) = min(D, scale_height(k)*csound(k))
            if (dbg) write(*,2) 'D_DSI', k, s% q(k), &
               s% D_DSI(k), D, scale_height(k)*csound(k), &
               Ri(k), Ri_crit, height, t_dyn(k), &
               instability_height, scale_height(k), csound(k)
         end do
         if (dbg) write(*,*)
      end if
   end do
   if (dbg) stop 'set_D_DSI'
end subroutine set_D_DSI
```

■ /private/rotation_mix_info.f

```
subroutine set_D_DSI(ierr)
   integer, intent(out) :: ierr
   integer :: i, k, kbot, ktop
   real(dp) :: instability_height, height, D
   logical, parameter :: dbg = .false.
   include 'formats.dek'

   ierr = 0
   kbot = nz
   do i = nz-1, 1, -1
      if (Ri(i) < Ri_crit .and. s% mixing_type(i) /= convective_mixing) then
         unstable(i_DSI,i) = .true.
         if (.not. unstable(i_DSI,i+1)) kbot = i
      end if
      if (unstable(i_DSI,i+1) .and. &
            (i == 1 .or. .not. unstable(i_DSI,i)) .and. kbot > 1) then
         if (unstable(i_DSI,i)) then
            ktop = i
         else
            ktop = i+1
         end if
         instability_height = r(ktop) - r(kbot)
         if (dbg) write(*,3) 'DSI: ktop, kbot', ktop, kbot
         do k = ktop, kbot
            height = min(instability_height, scale_height(k))
            D = s% D_DSI_factor*height**2/t_dyn(k)
            s% D_DSI(k) = min(D, scale_height(k)*csound(k))
            if (dbg) write(*,2) 'D_DSI', k, s% q(k), &
               s% D_DSI(k), D, scale_height(k)*csound(k), &
               Ri(k), Ri_crit, height, t_dyn(k), &
               instability_height, scale_height(k), csound(k)
         end do
         if (dbg) write(*,*)
      end if
   end do
   if (dbg) stop 'set_D_DSI'
end subroutine set_D_DSI
```

```
subroutine set_D_DSI(ierr)
   integer, intent(out) :: ierr
   integer :: i, k, kbot, ktop
   real(dp) :: instability_height, height, D
   logical, parameter :: dbg = .false.
   include 'formats.dek'

   ierr = 0
   kbot = nz
   do i = nz-1, 1, -1
      if (Ri(i) < Ri_crit .and. s% mixing_type(i) /= convective_mixing) then
         unstable(i_DSI,i) = .true.
         if (.not. unstable(i_DSI,i+1)) kbot = i
      end if
      if (unstable(i_DSI,i+1) .and. &
            (i == 1 .or. .not. unstable(i_DSI,i)) .and. kbot > 1) then
         if (unstable(i_DSI,i)) then
            ktop = i
         else
            ktop = i+1
         end if
         instability_height = r(ktop) - r(kbot)
         if (dbg) write(*,3) 'DSI: ktop, kbot', ktop, kbot
         do k = ktop, kbot
            height = min(instability_height, scale_height(k))
            D = s% D_DSI_factor*height**2/t_dyn(k)
            s% D_DSI(k) = min(D, scale_height(k)*csound(k))
            if (dbg) write(*,2) 'D_DSI', k, s% q(k), &
               s% D_DSI(k), D, scale_height(k)*csound(k), &
               Ri(k), Ri_crit, height, t_dyn(k), &
               instability_height, scale_height(k), csound(k)
         end do
         if (dbg) write(*,*)
      end if
   end do
   if (dbg) stop 'set_D_DSI'
end subroutine set_D_DSI
```

Dynamical Shear (D_DSI)

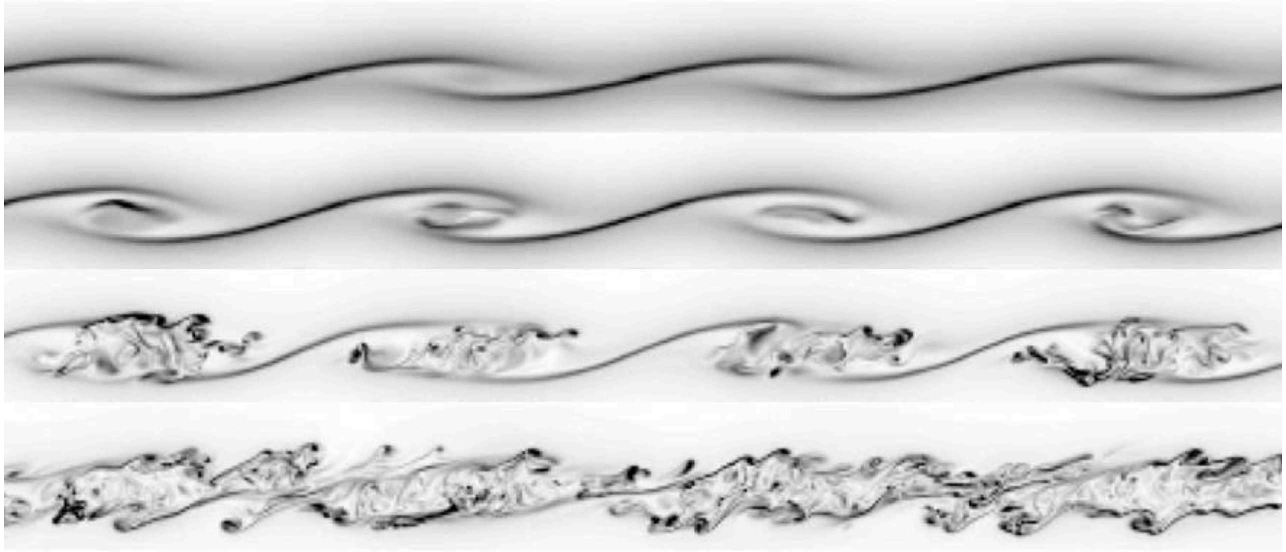
$$\mathcal{R}i \equiv \frac{N^2}{(dV/dz)^2} < \mathcal{R}i_{\rm crit}, \quad \text{with}$$

$$N^2 = \frac{g\delta}{H_P}\left(\nabla_{\rm ad} - \nabla + \frac{\varphi}{\delta}\nabla_\mu\right).$$

$$D_{\rm shear} = \frac{1}{3}v\ell = 2\mathcal{R}i_{\rm crit}K\frac{(dV/dz)^2}{N^2}$$

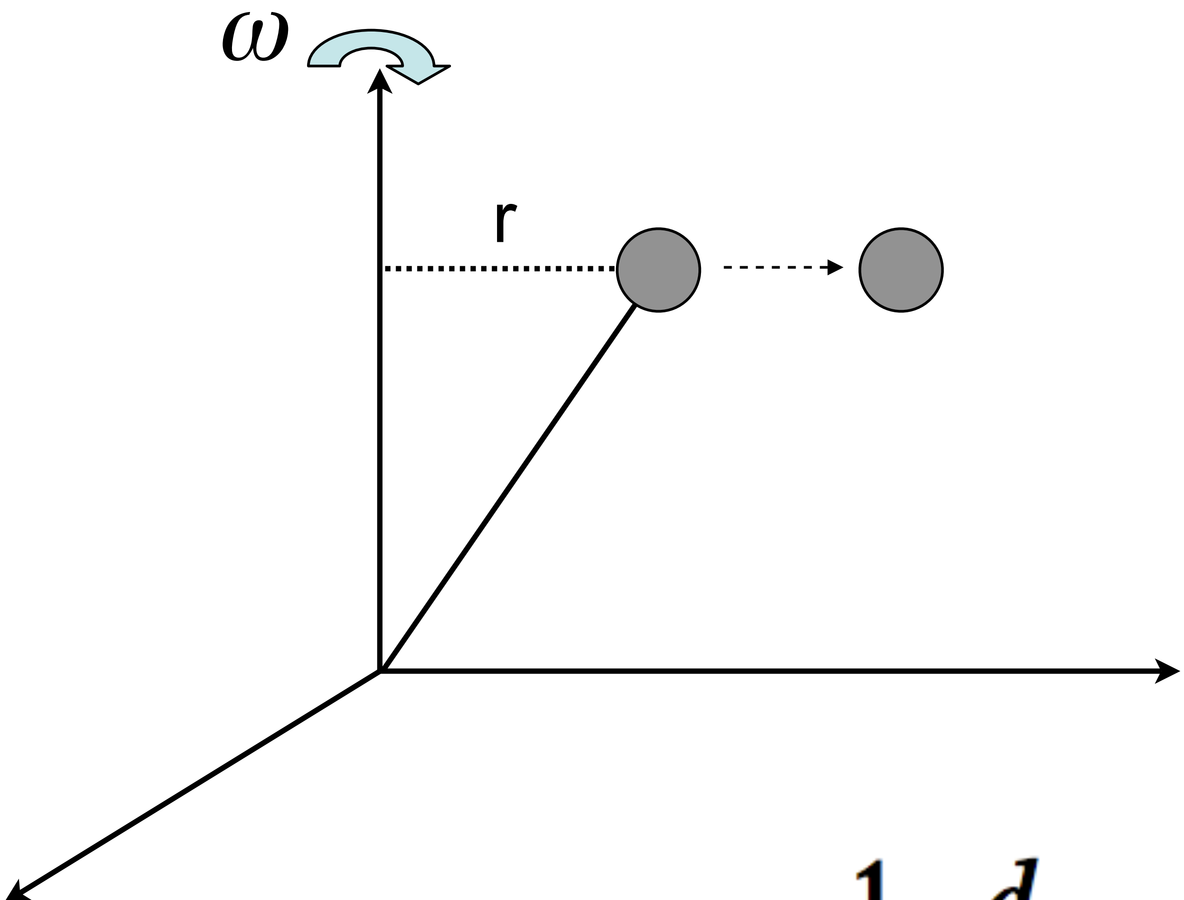
Secular Shear (D_SSI)

Thermal Timescale



- In the presence of a stabilizing thermal gradient, an eddy might have to wait for heat to diffuse out before an overturn is energetically favorable
- Mixing process on t_{KH}
- Results in D_SSI, entering the diffusion equations

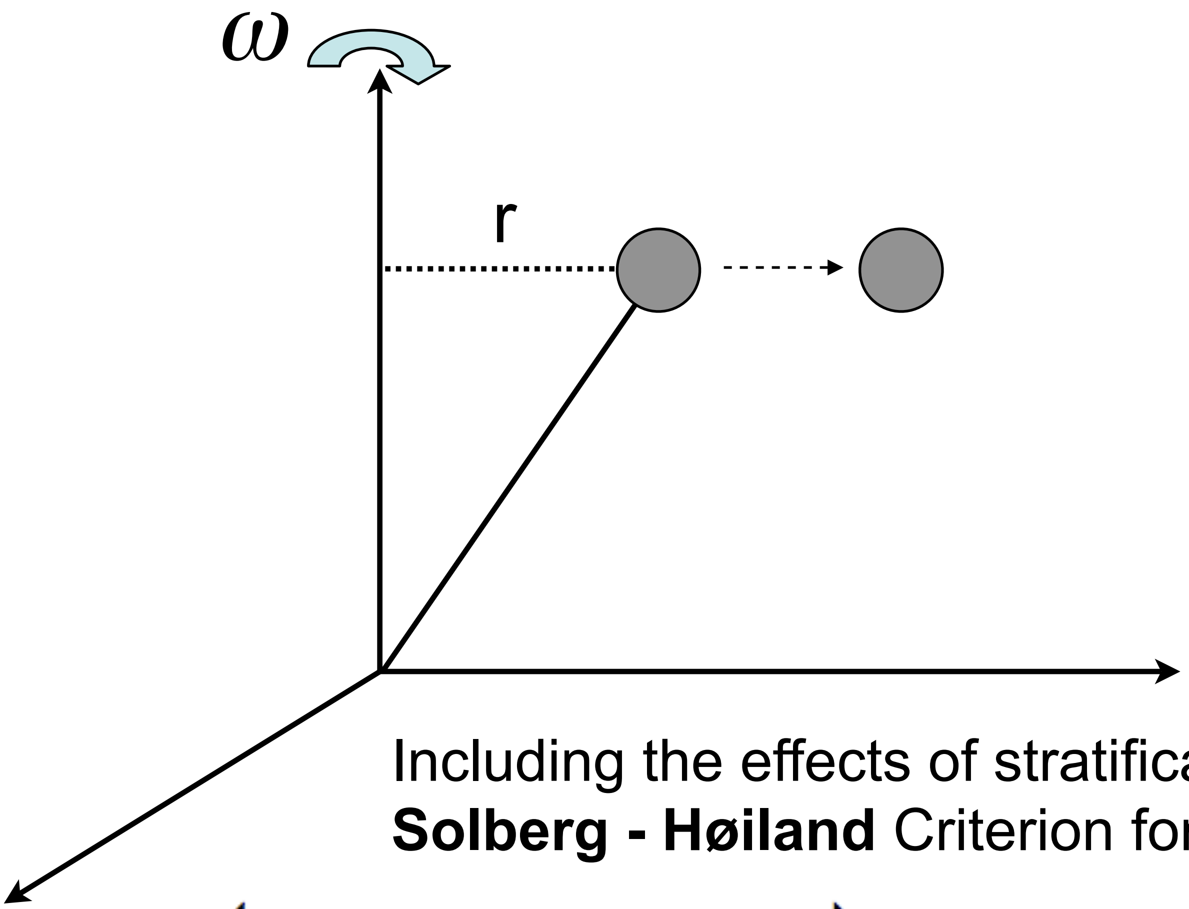
Solberg - Høiland (D_SH)



Rayleigh Criterion for stability

$$\frac{1}{r^3}\frac{d}{dr}(r^2\omega)^2 \geq 0$$

Solberg - Høiland (D_SH)

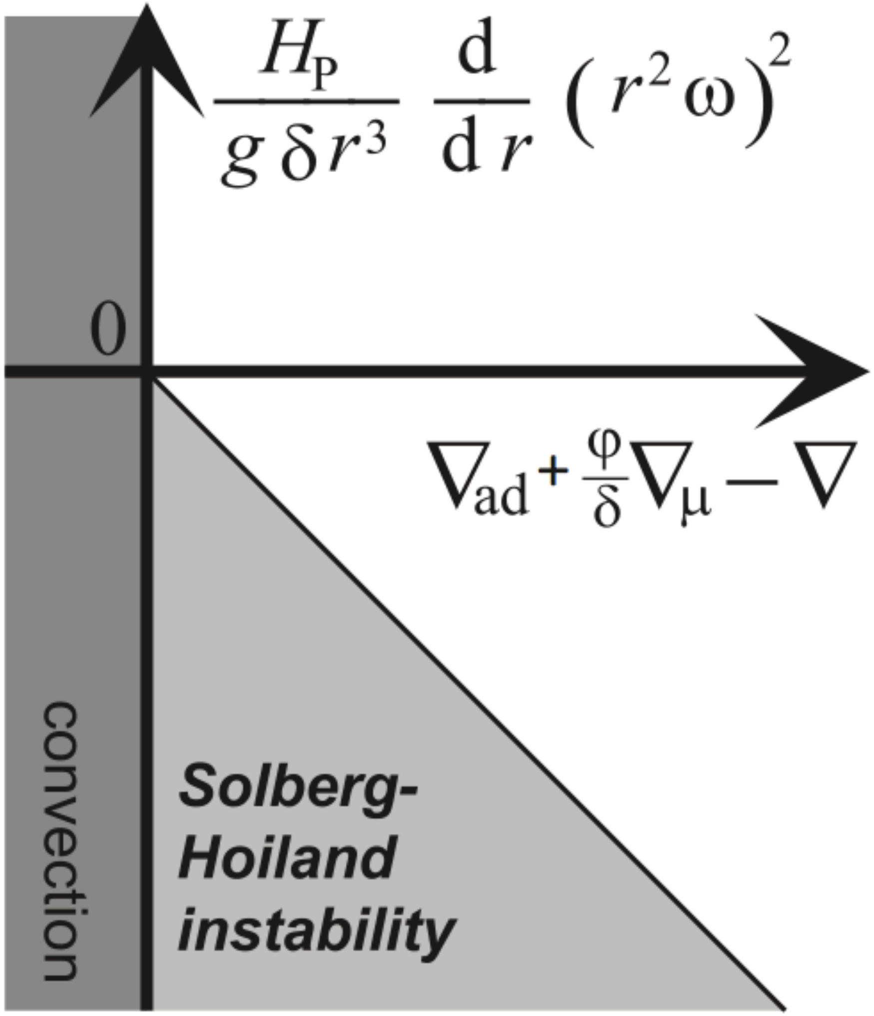


Including the effects of stratification etc.
Solberg - Høiland Criterion for stability

$$R_{\rm SH} \equiv \frac{g\delta}{H_P}\left(\nabla_{\rm ad} - \nabla + \frac{\varphi}{\delta}\nabla_\mu\right) + \frac{1}{r^3}\frac{d}{dr}(r^2\omega)^2 \geq 0$$

Solberg - Høiland (D_SH)

- For no rotation (or constant j) we recover Ledoux
- Can reduce size of convective regions (as j tends to increase outside, one has an extra restoring force)
- Results in D_SH, entering the diffusion equations

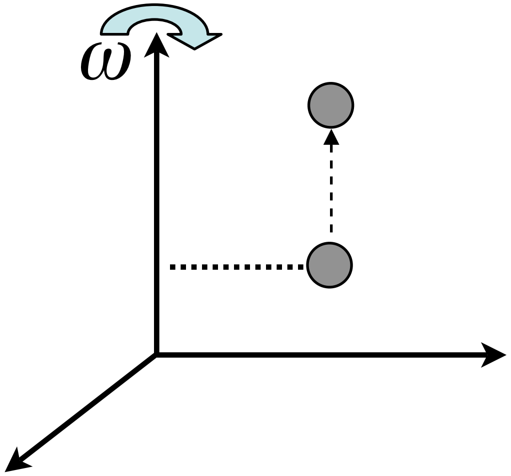


Heger et al. 2000

Goldreich-Schubert-Fricke (D_GSF)

- Baroclinic instability
- Two criteria: one is the secular analog of the SH, the other is an analog of the Taylor-Proudman theorem.
- The second criteria is generally in contradiction with the shellular law. GSF would try to enforce uniform rotation. D_GSF entering the diffusion equations

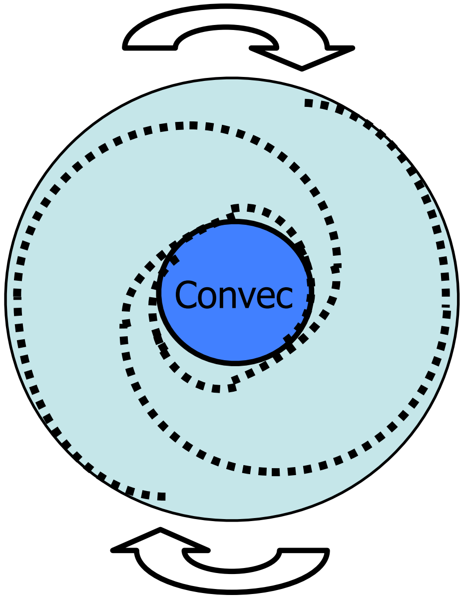
$$\frac{\partial j}{\partial r} \geq 0 \quad \text{and} \quad \frac{\partial \omega}{\partial z} = 0$$

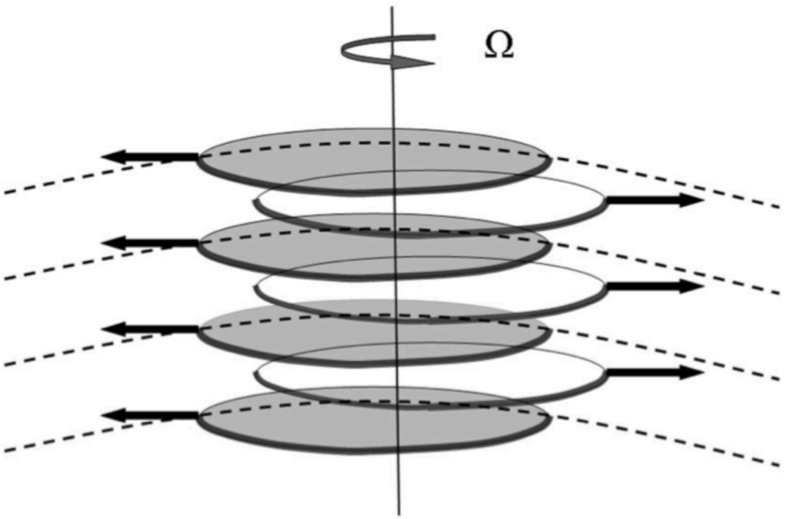


Spruit-Tayler (D_ST)

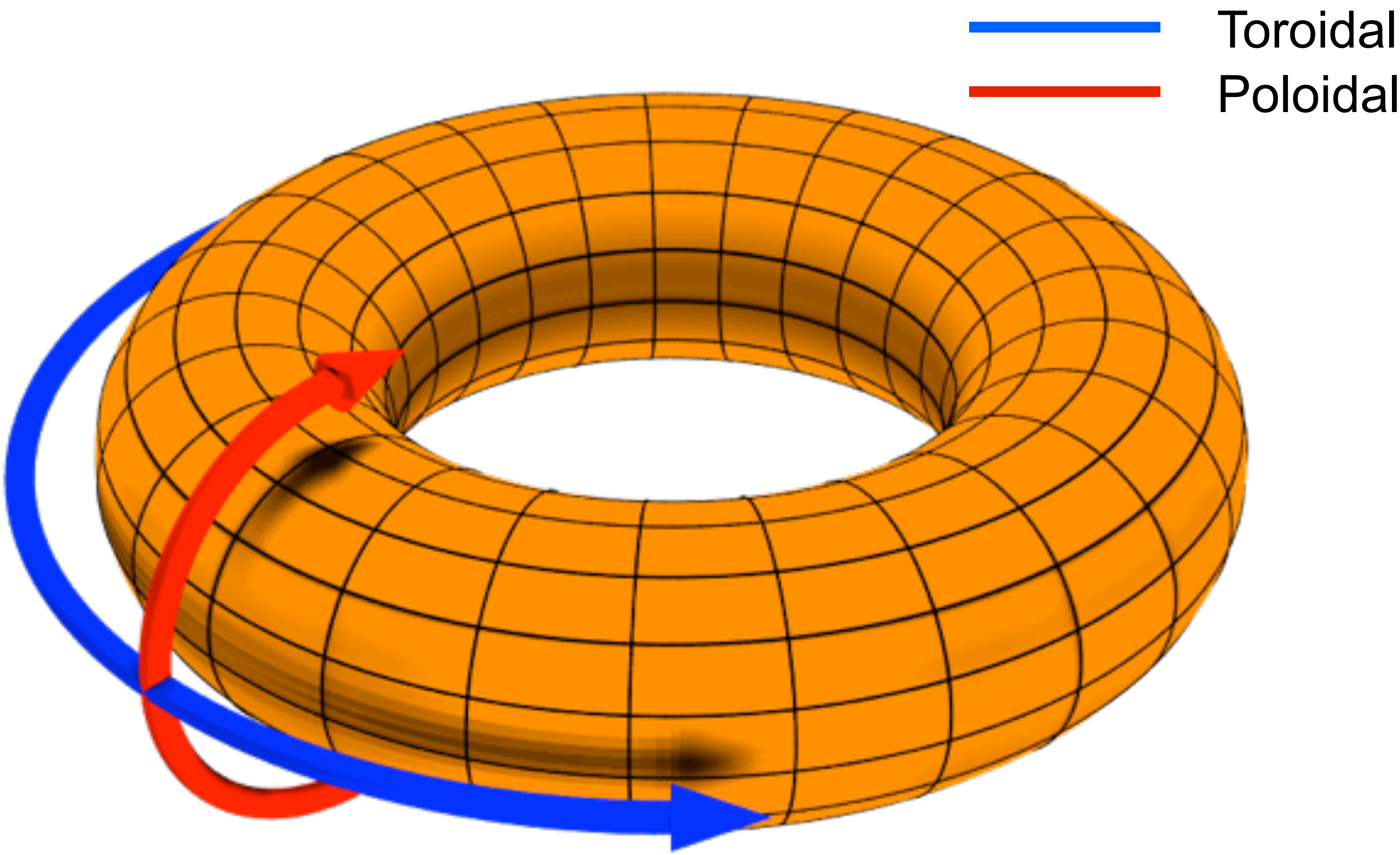
- Dynamo in a radiative layer
- Magnetic energy is generated from differential rotation
- Initially a seed magnetic field is stretched by the differential rotation, amplifying the toroidal component of the field
- An instability in the toroidal component of the field (Tayler instability) is used to close the dynamo loop

Spruit 2002

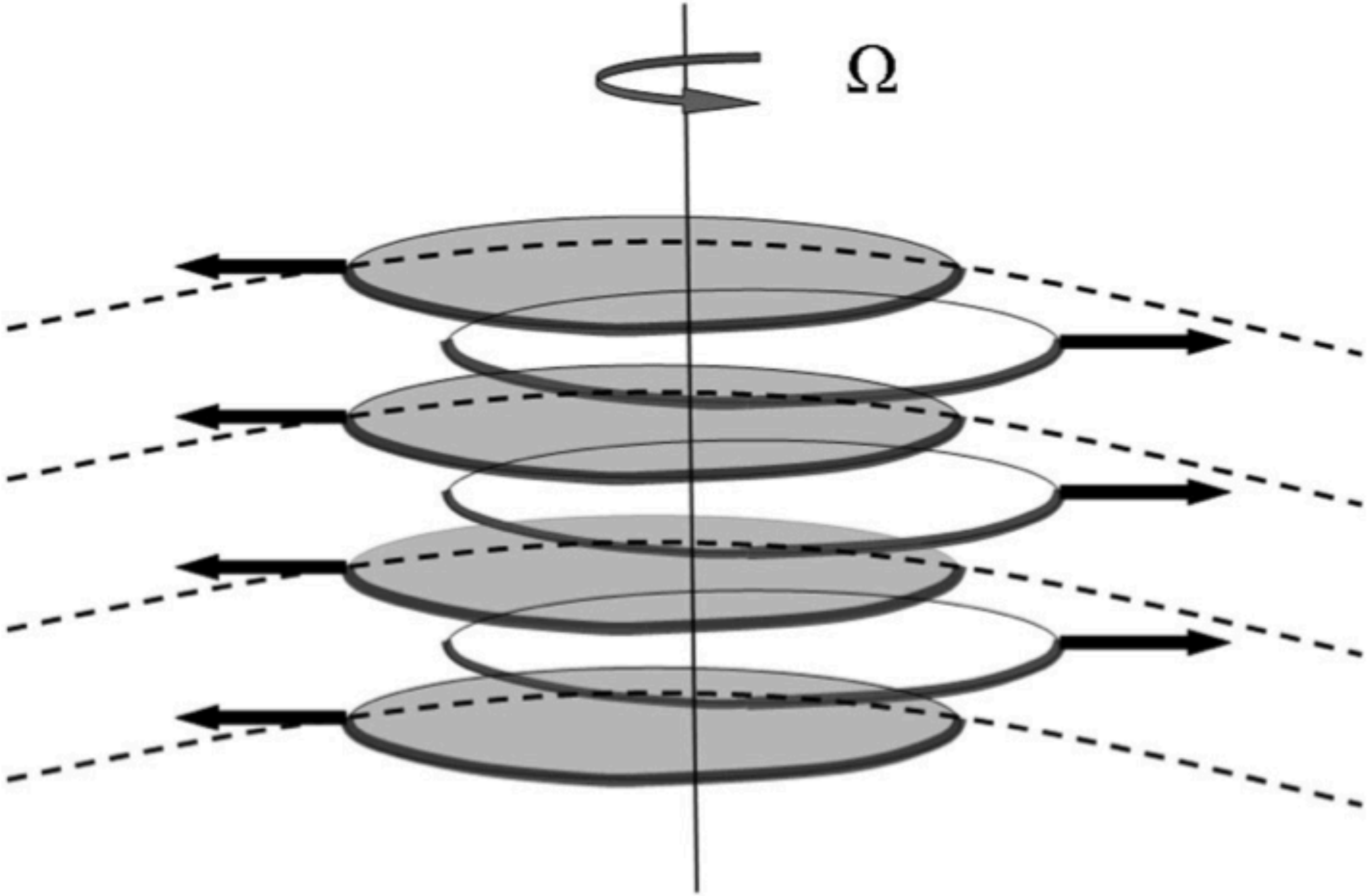




Spruit-Tayler (D_ST)



Tayler Instability

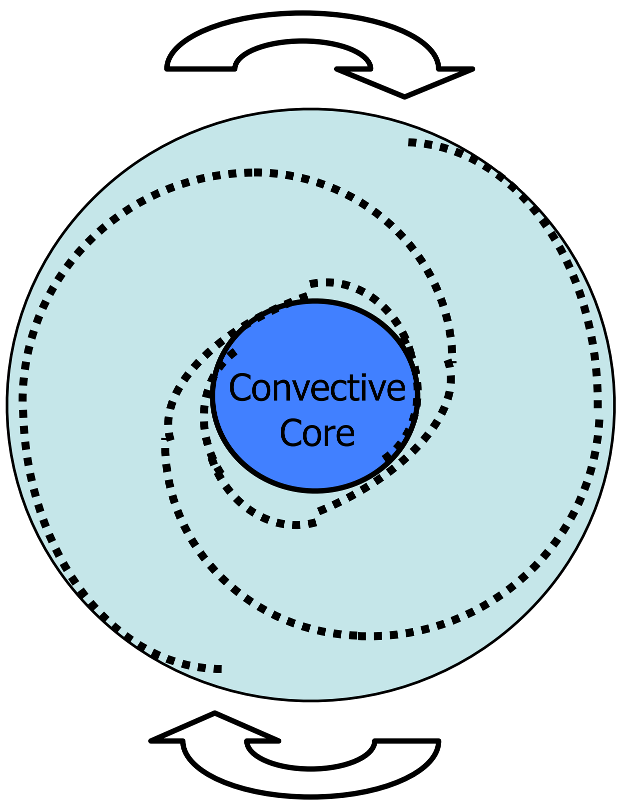


This is used to generate a poloidal component from the toroidal component of the field

Spruit 1999

Magnetic fields

- Spruit-Tayler Dynamo (Spruit 2002)
- Core - Envelope coupling



1. Differential rotation winds up toroidal component of B
2. Magnetic torques tend to restore rigid rotation

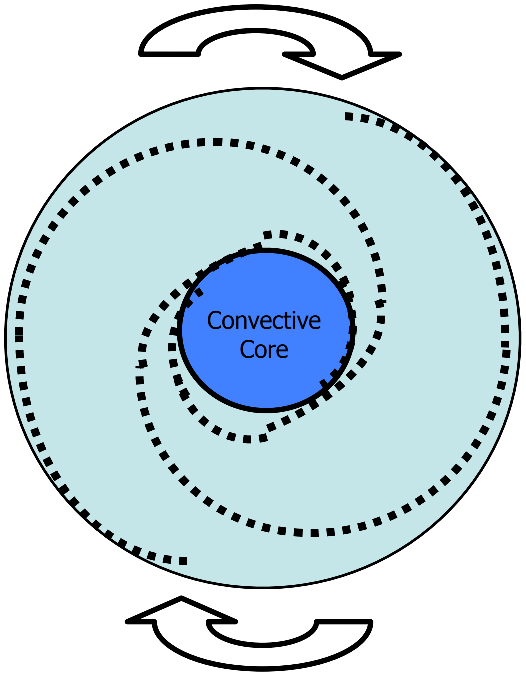
If the envelope slows down angular momentum is also removed from the core

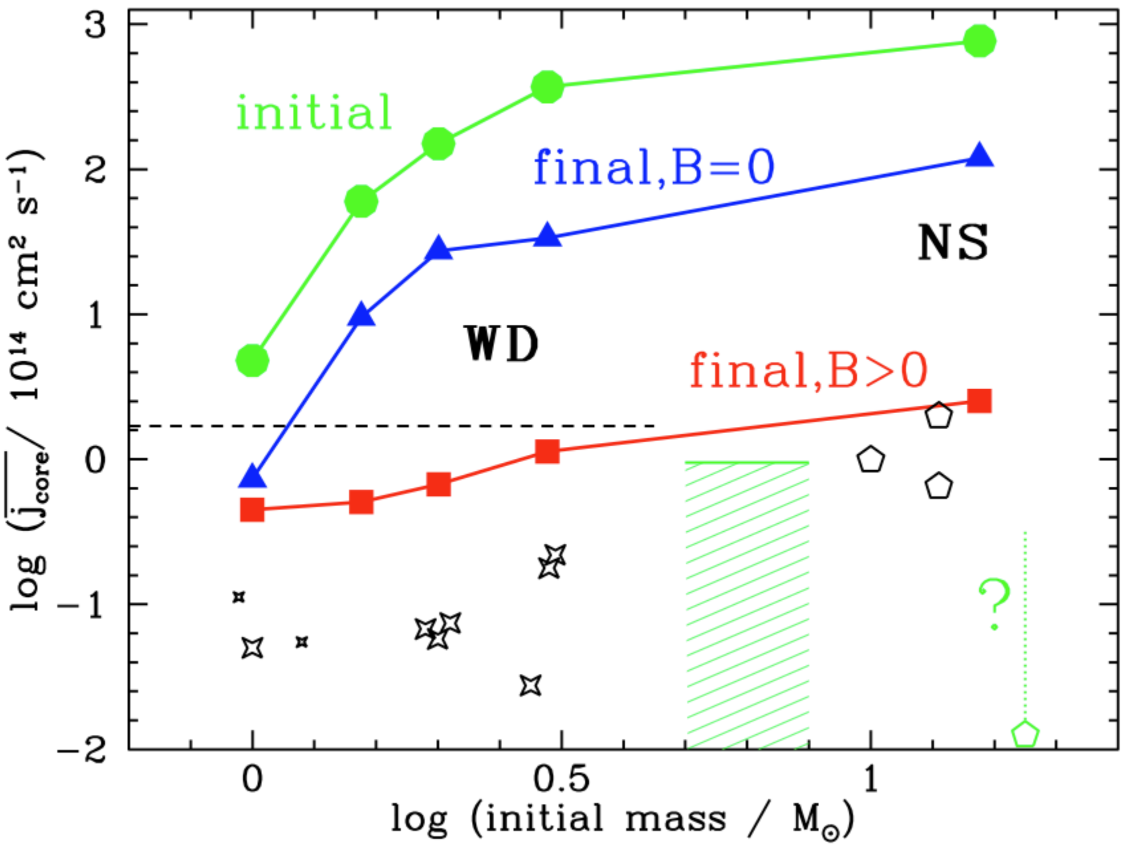
Debate on the ST Dynamo

- The ST dynamo is still under review
- While the Tayler instability is sound, the loop proposed by Spruit has been criticized
- Simulations of Zahn et al. 2007 could not find dynamo action
- On the other hand simulations of Braithwaite et al. 2006 showed the Spruit-Tayler dynamo
- The jury is still out, but it looks like a j-transport mechanism similar to the ST has to work in stars to reproduce some observations (e.g. asteroseismology and spin rates of compact remnants. g-waves could also play a role)

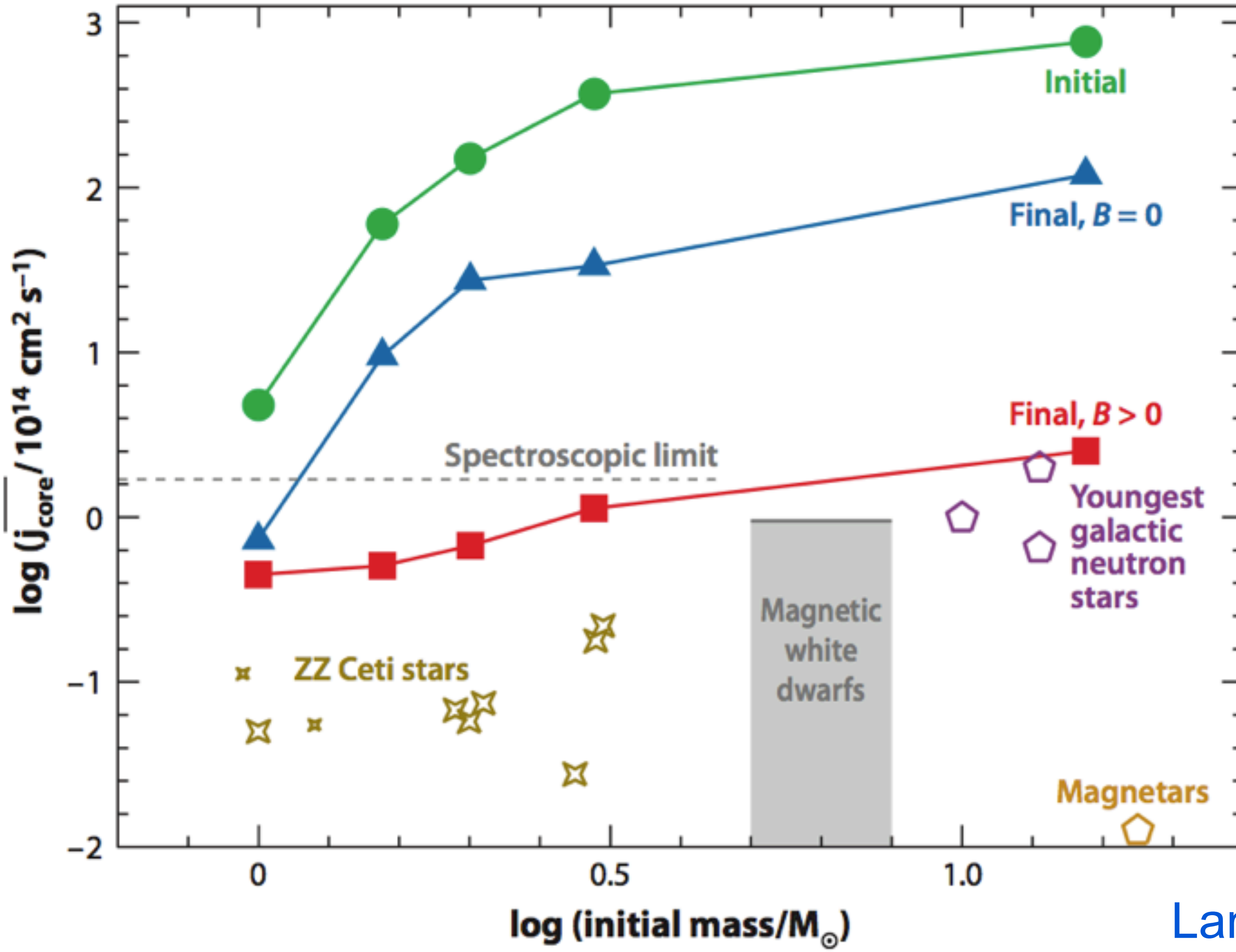
Magnetic fields

- Spin of compact remnants
  (Heger et al. 2005, Suijs et al. 2008)
  - Core-envelope coupling leads to spin-down of the core





Need for an extra j-transport mechanism



Langer 2012

Turning on and off the TS fields in MESA

```
! set to 0 for non-magnetic
         D_ST_factor = 1.0
         nu_ST_factor = 1.0
```

$$\left(\frac{\partial X_n}{\partial t}\right)_m = \left(\frac{\partial}{\partial m}\right)_t \left[(4\pi r^2 \rho)^2 D \left(\frac{\partial X_n}{\partial m}\right)_t\right] + \left(\frac{\mathrm{d}X_n}{\mathrm{d}t}\right)_{\mathrm{nuc}}$$

$$\left(\frac{\partial \omega}{\partial t}\right)_m = \frac{1}{i}\left(\frac{\partial}{\partial m}\right)_t \left[(4\pi r^2 \rho)^2 i \nu \left(\frac{\partial \omega}{\partial m}\right)_t\right] - \frac{2\omega}{r}\left(\frac{\partial r}{\partial t}\right)_m \left(\frac{1}{2}\frac{d \ln i}{d \ln r}\right)$$

and now... Mini Lab!