Surface-averaged gravity darkening corrections for 1-D stellar evolution models

Aaron Dotter

October 4, 2017

Abstract

Rotating stars are subject to gravity darkening, in which the flux at the stellar surface is proportional to the local surface gravity and, thus, a rotating star appears both hotter and brighter at the poles than at the equator. This document describes a method to derive gravity darkening corrections that can be applied to stellar evolution models; these adjust the intrinsic model effective temperature , surface gravity, and luminosity in order to account for the effects of surface rotation as well as the angle between the stellar rotation axis and the line of sight. I provide a Python implementation to apply these corrections to stellar evolution models.

1 The gravity darkening model

The gravity darkening model is derived by Espinosa Lara & Rieutord (2011, A&A, 533, 43; hereafter ELR). ELR assume the radiative flux is directed antiparallel to the effective surface gravity. The effective surface gravity is not uniform in a rotating star and, thus, neither is the flux. Since the scalar flux (F) is related to the effective temperature (T_{eff}) by the Stefan-Boltzmann law, the same argument applies to both F and T_{eff}. The ELR model reduces to a differential equation characterized by two variables: the polar angle θ and the ratio of surface angular velocity to the Keplerian angular velocity (ELR eq. 10):

$$\omega = \frac{\Omega}{\Omega_K} = \Omega\sqrt{\frac{R_e}{GM}} \tag{1}$$

where Ω is the surface angular velocity (radians per second, assumed uniform), Ω_K is the Keplerian angular velocity, R_e is the equatorial radius of the star, G is Newton's gravitational constant, and M is stellar mass.

The ELR model is valid for $0 \leq \theta \leq \pi/2$; all other values of θ are mapped into this interval via symmetry arguments. The model requires a numerical solution, except at the extrema $\theta = 0$ (ELR eq. 27) and $\pi/2$ (ELR eq. 28) for which analytical solutions are provided. The numerical solution boils down to finding the value of $\tilde{r} = R/R_e$ that satisfies (ELR eq. 30)

$$\frac{1}{\omega^2 \tilde{r}} + \frac{1}{2}\tilde{r}^2 \sin^2\theta = \frac{1}{\omega^2} + \frac{1}{2} \tag{2}$$

for given ω and θ. The values of $\tilde{r}$, θ, and ω are then used to solve for the modified angular variable ϑ (ELR eq. 24)

$$\cos\vartheta + \ln\tan(\vartheta/2) = \frac{1}{3}\omega^2\tilde{r}^3\cos^3\theta + \cos\theta + \ln\tan(\theta/2) \tag{3}$$

These are straightforward to solve with a root-find and, at this point one can calculate the normalized temperature variable $\tilde{\mathrm{T}}_{\mathrm{eff}} = \mathrm{T}_{\mathrm{eff}}\left(\frac{L}{4\pi\sigma R_e^2}\right)^{-1/4}$ via ELR eq. 31.

Figure 1 shows the solution of the ELR model as a function of θ for different values of ω. Figure 1 indicates that the polar radius reaches a minimum value of $R_p = (2/3)R_e$ at $\omega = \pi/2$. $\tilde{\mathrm{T}}_{\mathrm{eff}}$ is included in the right panel of Figure 1. When $\omega = 0$, the idealized star is spherical and, thus, $R_p = R_e$ and $\tilde{\mathrm{T}}_{\mathrm{eff}} = \mathrm{T}_{\mathrm{eff}}$ for all θ.

2 Oblate spheroids and projection effects

Now that we have a solution for the scaled flux ($\tilde{F}$) we can compute the projected luminosity along the line of sight (LOS). The surface is an oblate spheroid so we work in oblate spheroidal coordinates. The coordinate system is defined by three variables: polar angle ν, azimuthal angle ϕ, and μ = arctanh(R_p/R_e). It is important to distinguish between ν in the oblate spheroidal coordinate system and the polar angle θ in the ELR model: $\nu = 0$ at the equator; $\nu = \pi/2$ at the pole and, thus, $\theta = \pi/2 - \nu$.

With the oblate spheroidal coordinate system thus defined, and properly connected to the ELR model, we can now integrate over the stellar surface

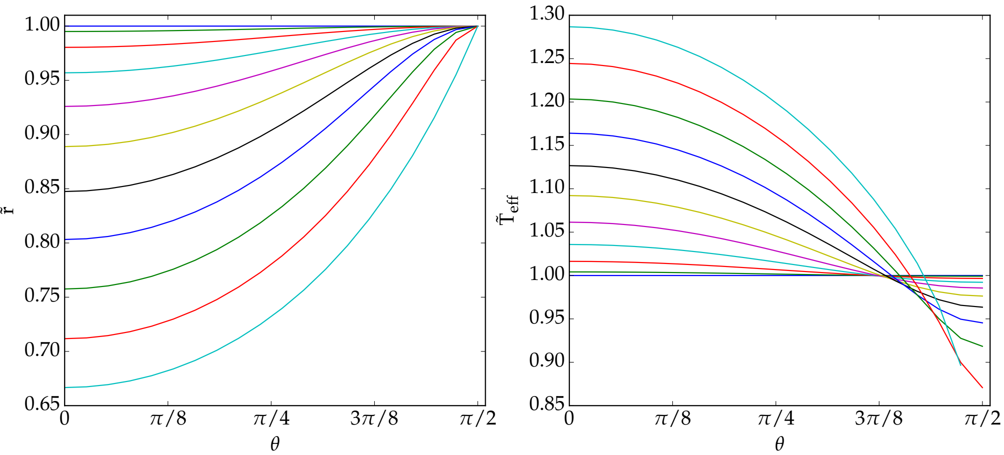

Figure 1: *Left:* The ratio of the stellar radius to the equatorial radius ($\tilde{r} = R/R_e$) as a function of the polar angle θ for different values of ω. *Right:* Similar to the left panel but now showing $\tilde{\mathrm{T}}_{\mathrm{eff}}$, see text for details.

Σ. However, the quantity that we want is that projected along the line of sight (LOS) and for that we need to define one additional angle i and the LOS unit vector $\hat{l}$. i is chosen such that $i = 0$ at the equator. The projected surface is denoted by Σ_{proj}.

A useful analytical formula to calculate Σ_{proj} is given by Binngeli (1980). Note that Brandt & Huang (2015) use a different formula, which is only (trivially) correct at $i = 0$ and $\pi/2$. The Brandt & Huang formula leads to a maximum error of $\sim 8\%$ in Σ_{proj} at $i = \pi/4$.

To calculate the luminosity projected along the LOS L_{proj} requires the surface integral

$$L_{proj} = 4 \iint\limits_{d\vec{\Sigma}\cdot\hat{l}>0} F d\vec{\Sigma} \cdot \hat{l} \tag{4}$$

where only the flux projected toward the observer, i.e., $d\vec{\Sigma} \cdot \hat{l} \geq 0$, is kept. Once L_{proj} and Σ_{proj} are known, the projected effective temperature, $\mathrm{T}_{\mathrm{eff}\,proj}$, can be obtained from the Stefan-Boltzmann law

$$\mathrm{T}_{\mathrm{eff}\,proj} = \left(\frac{L_{proj}}{\sigma \Sigma_{proj}}\right)^{1/4} \tag{5}$$

where σ is the Stefan-Boltzmann constant.

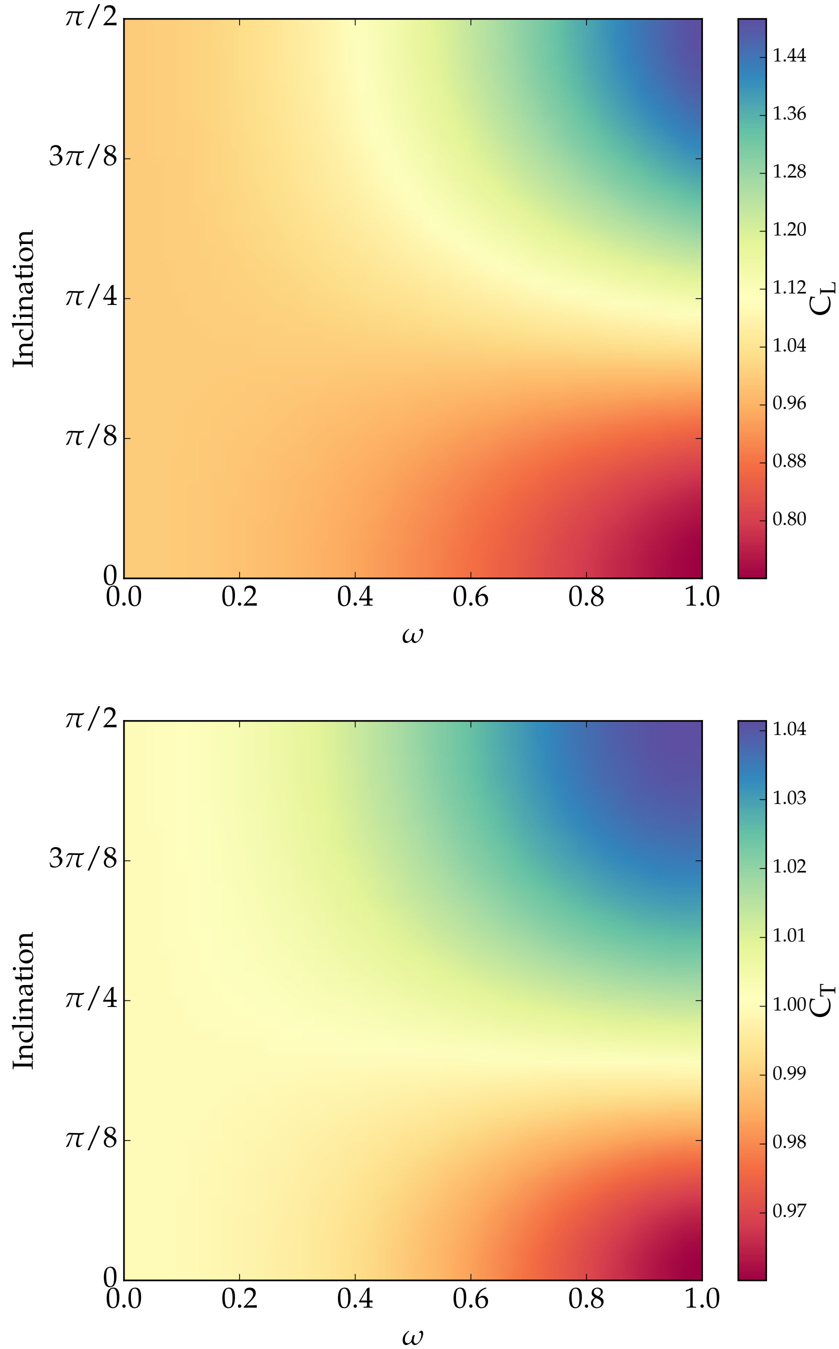

Figure 2: The range of C_L (upper panel) and C_T (lower panel). Note the color scale is different in the two panels.

The following is based closely on what is done in the SYCLIST code (Georgy et al. 2014, A&A, 566, 21). The double integral (4) is, essentially, a geometric factor that does not depend on the stellar temperature or luminosity. It only depends upon ω and i. Thus, it makes sense to derive correction factors which allow the projected quantities to be determined from the model intrinsic quantities by geometric scaling factors, i.e.,

$$L_{proj}(\omega, i) = C_L(\omega, i)L \tag{6}$$

and

$$\mathrm{T_{eff}}_{proj}(\omega, i) = C_T(\omega, i)\mathrm{T_{eff}} \tag{7}$$

Thus defined, the geometric factors C_L and C_T are given by

$$C_L = \frac{4 \iint\limits_{d\Sigma\cdot l>0} F d\vec{\Sigma} \cdot \hat{l}}{\iint\limits_{\Sigma} F d\Sigma} \tag{8}$$

and

$$C_T = \left(\frac{C_L \Sigma}{4\Sigma_{proj}} \right)^{1/4} \tag{9}$$

Figure 2 shows how the values of C_T and C_L vary over the $omega$,i-plane. These can be compared with the upper panels of Figure XXX in Georgy et al. (2014). The variation in C_L is greater than that of C_T due to the 1/4-power relationship between the two. At $\omega = 1$, where the geometric factors change the most, C_L varies from -25% to $+50\%$ while C_T varies by $\pm 4\%$. When the LOS is oriented at $i = 34°$ (0.6 rad) above the equatorial plane, both C_L , $C_T \simeq 1$ for all ω.

3 Python code for C_T and C_L

The github repository provides Python code to numerically solve the differential equations in the ELR model and to compute the geometric coefficients C_L and C_T for arbitrary ω and i. The repository also includes a Python interface to interpolate C_L and C_T coefficients from a 50×50 grid stored in a data file. Separate interpolants are provided for C_L and C_T to enable the application of gravity darkening corrections to stellar evolution models. As an example, Figure 3 shows a MIST isochrone (Choi et al. 2016, ApJ, 823, 102) with

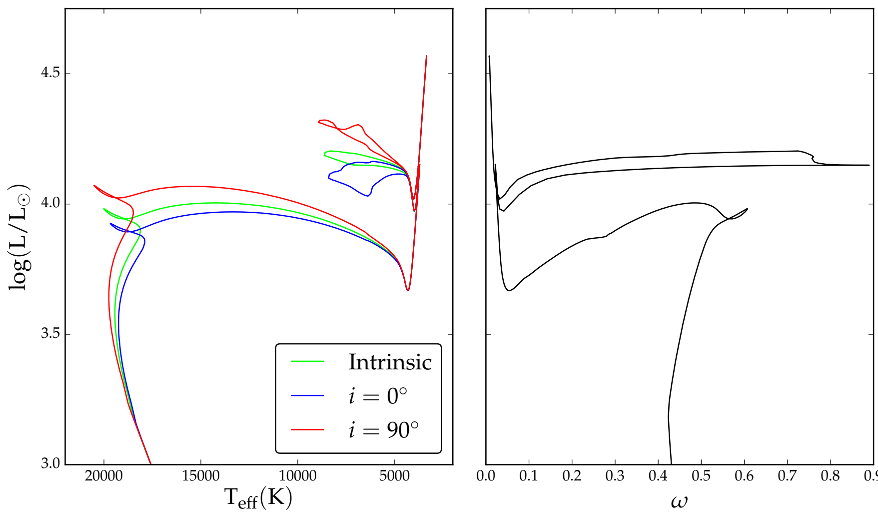

Figure 3: MIST isochrone with [Fe/H]=0, log(age[yr])=7.5, and $\omega = 0.4$ at the ZAMS. *Left:* The H-R diagram showing the effect of changing the inclination angle i from 0° (in the plane of equator) to 90° (aligned with the rotation axis). *Right:* The variation in ω with luminosity to explain why the gravity darkening correction is larger or smaller at a given point on the isochrone in the H-R diagram.

$\omega = 0.4$ (at the ZAMS). The left panel shows the intrinsic $T_{\rm eff}$ and luminosity as well as the gravity-darkened values for $i = 0$ and $i = 90°$. The right panel shows how ω evolves along the isochrone from the initial value of $\omega = 0.4$ set at the ZAMS. The most dramatic differences in the H-R diagram occur when ω is large. ω is provided in MIST as `surf_avg_omega_div_omega_crit`.