Gauge Theory Prospects

β Function Analysis

Abstract

The experimental value 125.35 GeV for the Higgs mass hints at the possibility of a new gauge theory. This theory is postulated to be asymptotically free and similar in nature to QCD. This paper takes a look at models with a varying number of constituents or flavors. The objective is to discredit models that contain infrared (or ultraviolet) fixed points. Perturbative models with a number of constituents from 8 to 12 have long been a source of controversy. The question remains as to if there is a better approximation scheme to determine whether there are infrared (or ultraviolet) fixed points especially for the 5 loop β function. The Pade approximation has merit as a better approximation scheme, however varying L and M values and poles actually increase ambiguity. More specifically Pade approximations $\beta_{4,2}$ and $\beta_{3,3}$ predict an ultraviolet fixed point at $a_s = 0.20$ for number of flavors $n_f = 12$ for the 5 loop β function.

Author: Justin Rivers

Advisor: Dr. Kieran Holland

Date: 26 May, 2020

Introduction

Protons and neutrons are considered the building blocks of normal matter. In the 1960s, it was found that these larger hadron particles behaved like excitations of yet more fundamental particles. These particles are known today as quarks. 6 different types or flavors of quarks and their antiquarks have been discovered. These are the up, down, strange, charm, bottom and top quarks. Combinations of these quarks and their anti-quark partners make up not only protons and neutrons but a whole range of larger hadron particles. These particles (along with another family of particles called leptons) and the ways in which they interact make up what is known as the Standard Model of particle physics.

Quarks exhibit several strange characteristics. Each quark has one third of a charge called color charge [8](pg. 593). These colors are named red, green, and blue after the primary colors, although there is no relation. Quarks are held together by the strong nuclear force which is mediated by a massless particle known as a gluon. Gluons themselves can actually carry two color charges. Strangely, gluons hold quarks together in such a way that the strength of the quark-quark interactions actually grows as the distance between them grows. In quantum field theory, distance is related to energy through the Hamiltonian. Quarks are trapped together and at some point if they are pulled apart with enough distance, it is actually more energetically favorable to create a quark-antiquark pair that is bound to the original quarks. This is what is known as color confinement. The converse is also true, that at smaller distances the interaction is less. This downward trend in interaction (known as coupling) with increased proximity is called asymptotic freedom. The theory that describes the physical laws governing the strong nuclear force in this way is Quantum Chromodynamics (QCD).

QCD is an asymptotically free non-abelian gauge theory with SU(3) symmetry. Gauge theories have been extremely powerful and accurate in describing some of our more fundamental forces since the advent of Quantum Electrodynamics (QED) which describes electromagnetism. Although gauge theories are beyond the scope of this essay, I feel it is necessary to give a little explanation for further research purposes. A gauge theory itself is a theory that adopts a certain quantity, usually potential as a gauge which leaves certain quantities unchanged in the event of local transformations. The quantities that are unchanged are called invariant and they are chosen to be invariant to preserve some sort of physical symmetry. The process of determining the gauge is called gauge fixing and in QCD it is a fairly complicated process, but the general idea can be seen more readily in QED.

Maxwell's equations can be rewritten in the form of vector and scalar potentials rather than in terms of electric and magnetic fields and respective charge or current distributions.

$$\nabla^2 V + \frac{\partial}{\partial t}(\nabla \cdot \mathbf{A}) = -\frac{\rho}{\epsilon_0} \tag{1}$$

$$\left(\nabla^2 \mathbf{A} - \mu_0 \epsilon_0 \frac{\partial^2 \mathbf{A}}{\partial t^2}\right) - \nabla\left(\nabla \cdot \mathbf{A} + \mu_0 \epsilon_0 \frac{\partial V}{\partial t}\right) = -\mu_0 J \tag{2}$$

where the charge density ρ and current density J are in terms of A_k where $k = 1, 2, 3$ and V. These are vector and scalar potentials respectively. This follows from plugging into Maxwell's equations,

$$\mathbf{E} = -\nabla V - \frac{\partial \mathbf{A}}{\partial t} \tag{3}$$

$$\mathbf{B} = \nabla \times \mathbf{A} \tag{4}$$

Closer inspection of equations 3 and 4 reveal that E and B do not necessarily define the values of V and A_k. In fact, if we add a function $f(x, t)$ to A_k and then take it away from V, we will preserve the values of E and B. This means that the electric and magnetic fields remain invariant. The addition of these terms to the potentials are what are called gauge transformations. Here, gauge transformations allow us to pick a convenient term for the divergence of A_k, namely the Lorenz gauge

$$\nabla \cdot \mathbf{A} = -\mu_0 \epsilon_0 \frac{\partial V}{\partial t} \tag{5}$$

This allows for simplification of equations 1 and 2. See Chapter 10 in [7] for a more thorough explanation of invariance and gauge fixing for electrodynamics. In theories like QCD and QED the quantity that is truly meant to be invariant is the Lagrangian. All the possible transformations between different gauges form a group known as a symmetry or Lie group. In order to progress from one member of this group to another there is a set of mathematical rules called a Lie algebra. If these Lie groups are non-commutative, the theory is said to be non-abelian as in QCD. Vector fields arise from the Lie algebra of the symmetry group. The QED symmetry group is U(1) and is comprised of the electromagnetic 4-potential

$$A_\mu = (V/c, A_x, A_y, A_z)$$

Quantization in gauge theories leads to gauge bosons. The gauge bosons for QED and QCD are the photon and the gluon respectively. In gauge theories these types of gauge bosons interact with each other and constituent matter particles to create fields through a process known as coupling. The coupling in QED gets weaker at long distance and lower energies. In QCD the opposite happens, at smaller distance and higher energies the coupling weakens. This is asymptotic freedom and color confinement. Configurations of quarks undergoing confinement become the makeup of larger hadron particles. The mathematics

that define the coupling can actually be shown visually via Feynman diagrams. An example from QED can be found in figure 1. Feynman diagrams are then used to calculate different facets of the interaction. Scattering matrices, momentum, coupling and more can all be derived from the diagrams. The number of diagrams for interactions grow factorially with loop order. Loops are actually a phenomenon where an antiquark-quark pair are created and immediately destroyed. They are linked to the total momentum of the gauge boson and must be integrated over. There are about 100,000 diagrams just for loop order 5. Each coupling is an interaction with an elementary particle and its corresponding gauge boson. The analysis of the field strength is then reflected in the number of couplings and loops.

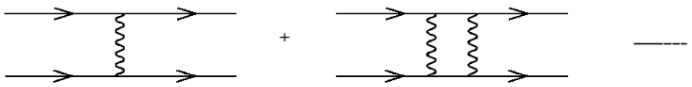


Figure 1: This is an example of two electrons scattering off of each other. The wavy lines are photons and the solid lines are electrons. They meet in vertices to which coupling is attributed to.

For any model, the β function describes the behavior of coupling as it pertains to energy scale.

$$\beta(a_s) = \mu^2 \frac{d}{d\mu^2} a_s(\mu) = -\sum_{i \geq 0} \beta_i a_s^{i+2} \tag{6}$$

Here, μ is the energy or mass scale. $a_s(\mu)$ is a function of coupling with

$$a_s = \frac{g(\mu)^2}{4\pi^2}$$

$g(\mu)$ is an analog to electric charge e in electromagnetism.

The loop orders β_i are functions of the number of constituent flavors of the corresponding model. I did not calculate these myself because according to [4] the 5 loop coefficient took some 20 years to obtain! I instead used the values from [3]:

$$\beta_0 = \frac{1}{4}\left\{11 - \frac{2}{3}n_f,\right\}, \quad \beta_1 = \frac{1}{4^2}\left\{102 - \frac{38}{3}n_f\right\},$$

$$\beta_2 = \frac{1}{4^3}\left\{\frac{2857}{2} - \frac{5033}{18}n_f + \frac{325}{54}n_f^2\right\},$$

$$\beta_3 = \frac{1}{4^4}\left\{\frac{149753}{6} + 3564\zeta_3 - \left[\frac{1078361}{162} + \frac{6508}{27}\zeta_3\right] n_f + \left[\frac{50065}{162} + \frac{6472}{81}\zeta_3\right] n_f^2 + \frac{1093}{729}n_f^3\right\},$$

$$\begin{aligned}\beta_4 = \frac{1}{4^5}\Bigg\{&\frac{8157455}{16} + \frac{621885}{2}\zeta_3 - \frac{88209}{2}\zeta_4 - 288090\zeta_5 \\ &+ n_f\left[-\frac{336460813}{1944} - \frac{4811164}{81}\zeta_3 + \frac{33935}{6}\zeta_4 + \frac{1358995}{27}\zeta_5\right] \\ &+ n_f^2\left[\frac{25960913}{1944} + \frac{698531}{81}\zeta_3 - \frac{10526}{9}\zeta_4 - \frac{381760}{81}\zeta_5\right] \\ &+ n_f^3\left[-\frac{630559}{5832} - \frac{48722}{243}\zeta_3 + \frac{1618}{27}\zeta_4 + \frac{460}{9}\zeta_5\right] \\ &+ n_f^4\left[\frac{1205}{2916} - \frac{152}{81}\zeta_3\right]\Bigg\},\end{aligned}$$

Higgs Mass and the need for New Gauge Theories

The Standard Model includes gauge theories that sum up the strong, weak, and electromagnetic forces. It accounts for and predicts particles that might arise from these gauge theories. One such interesting predicted particle is called the Higgs Boson which was found at CERN in 2013. The Standard Model has no explanation for the experimental value for the specific mass of the Higgs boson (125 GeV)[2]. This has prompted the scientific community to search for a more encompassing gauge theory framework that not only includes QCD but also adds to it in terms of such mass predictions.

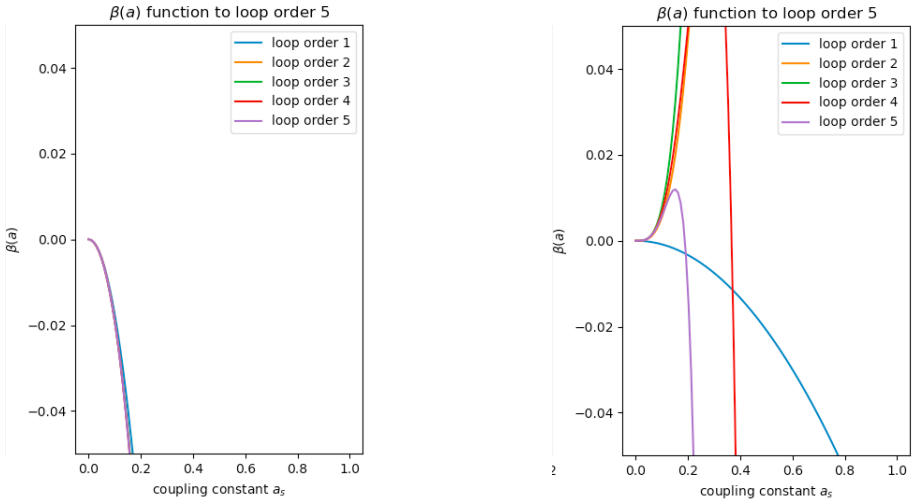
While only 6 flavors of elementary particles exist in the Standard Model via experimentation and observation, there has been extensive research lately into models with a higher number of flavors. Any plausible model will have its characteristic β function describing the coupling as a function of energy scale. It is theorized that any model that includes a β function that approaches zero would not produce the Higgs boson that we see today. This would mean that the mass scale of the constituent particles is 0 for that value of the coupling. It is hard to see how the Higgs mass could be dynamically generated by 0 mass constituents. Roots of the β function that approach zero from the negative side are called infrared fixed points. In contrast if the function approaches zero from

the positive side it is known as an ultraviolet fixed point. The goal is to find the fixed points and eliminate models that include them.

Perturbative methods

Coupling and the β Function

Using the definition of the β function in [3] (equation 6) as well as the coefficient values for up to 5 loop coupling, I have plotted the beta function out to an accuracy of 5 loops. The number of flavors varied from 1 to 16. After $n_f = 16$ the β function trends upward from zero which is no longer asymptotically free.



(a) $n_f = 6$ (b) $n_f = 16$

Figure 2: The graph of $n_f = 16$ shows the 5 loop in purple starting to trend upwards immediately when $a_s > 0$. For comparison, $n_f = 6$ starts and continues downward as a result of a negative β function. In between these values the function tends to start downward but in some cases approaches the horizontal axis. For models with certain values of n_f these are infrared fixed points. Technically $n_f = 16$ dips down slightly first giving an infrared and then an ultraviolet fixed point. Models after this do only trend upward. Lower loop orders are included for comparison.

The lower loop orders respond to a truncation in the series. For example, loop order 1 corresponds to a parabolic regime because the β function starts with a $b_0 a_s^2$ term. In many cases as n_f increases higher loop orders do not reflect small perturbations as one would expect to see in traditional power series. This is a cause for concern as the coefficients for the β function take a long time to calculate. It is possible a larger loop order could still be adding large

adjustments to the function so that below 5 loops the beta function is highly inaccurate. Physicists have found another way to describe coupling behavior through lattice gauge theory. Lattice gauge theory uses computer simulations and Monte Carlo methods to generate a discretized beta function which is then adjusted for lattice spacing. This paper will only cover the perturbative β function.

By varying n_f in integer steps I have found possible candidates for infrared fixed points at around $n_f = 12$ and $n_f = 13$. This seems to be consistent with findings in [4]. In figures 3 and 4 I have included models with $n_f = 12$ and $n_f = 13$ with the values of their roots. It is important to note that roots that are either negative or complex are unphysical. They violate unitarity in certain scattering matrices which leads to a lack of symmetry as well as loss of probability.

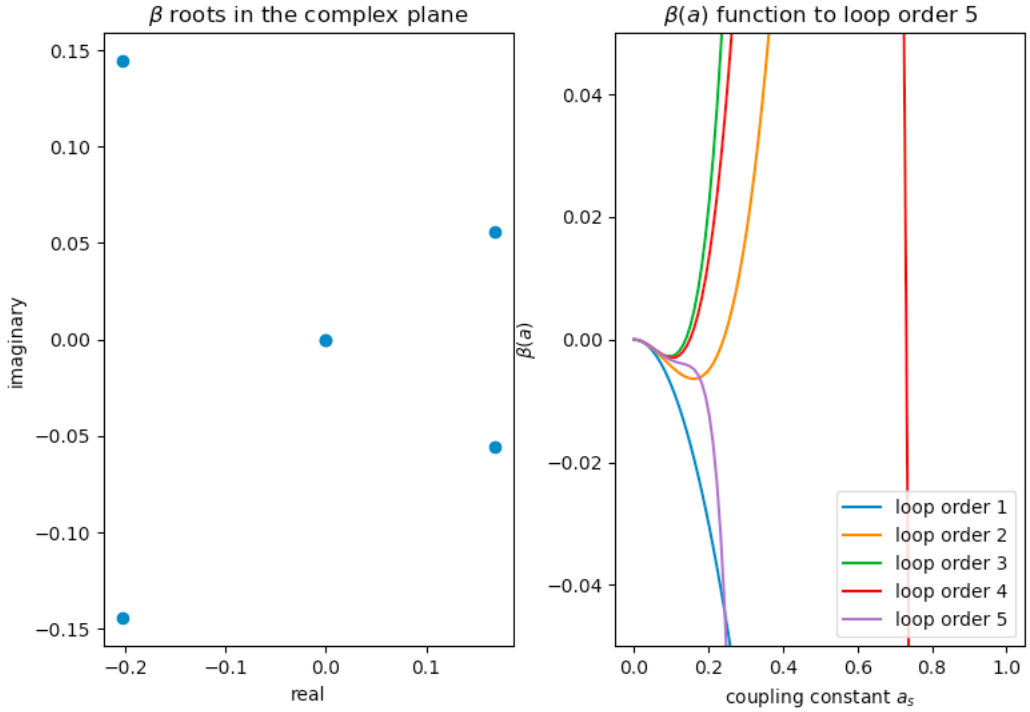


Figure 3: $n_f = 12$ β has no real roots: $(-0.203 - 0.144j)$ and conjugate, 0 multiplicity 2, and $(0.168 - 0.056j)$ and conjugate. Notice the purple loop order 5 dips and rises but not quite crossing the horizontal axis before trending downward again. (Left plot is loop order 5 roots only)

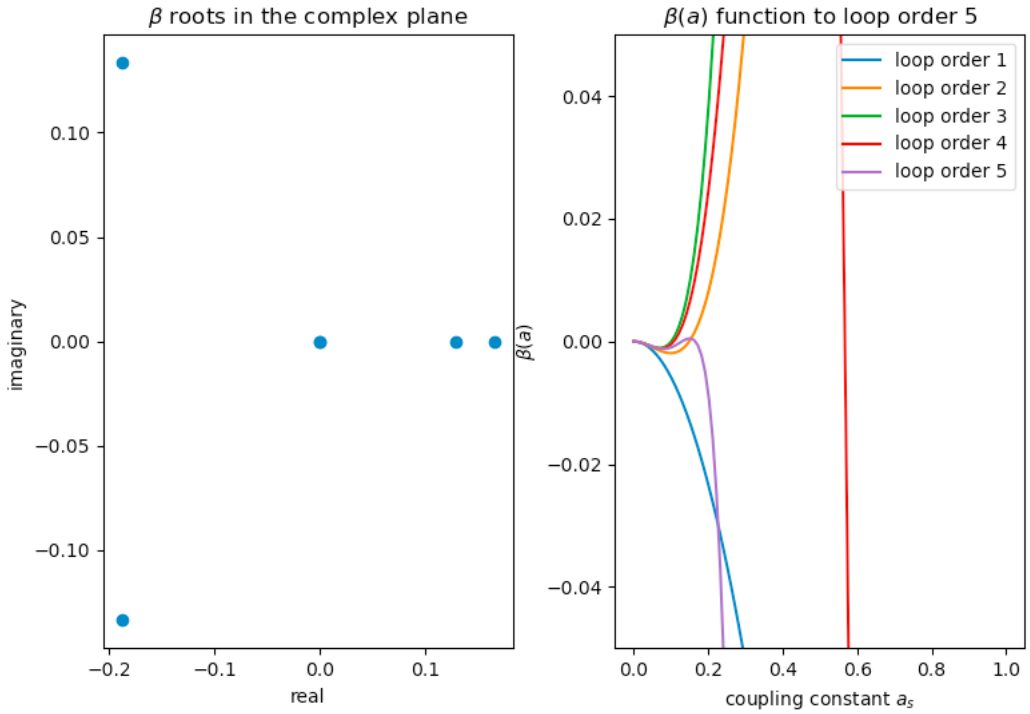


Figure 4: $n_f = 13$ β has 2 real roots: An infrared fixed point 0.129 and an ultraviolet fixed point at 0.165. Notice the purple loop order 5 dips and rises crossing the horizontal axis twice. (Left plot is loop order 5 roots only)

Further analysis of models with n_f values under 12 show that a positive pair of complex zeros seem to converge on the horizontal axis as n_f increases. In the figures above you can see the switch from complex roots in the 12 flavor model to two real roots in the 13 flavor model. After 13, the zeros remain real but again move apart. This suggests that there might be a zero value in between $n_f = 12$ and $n_f = 13$. Fractional flavors do not correspond to anything physical, but I decided to document the value for completeness. I allowed the values to range from 12 to 13 arbitrarily in 100 steps. There seemed to be a close match of multiplicity 2 around $n_f = 12.89$ shown in 5. Again this is consistent with behavior found in [4]. The value here would seem to be much closer to 13 possibly indicating that the 12 flavor model is fixed point free for the 5 loop order.

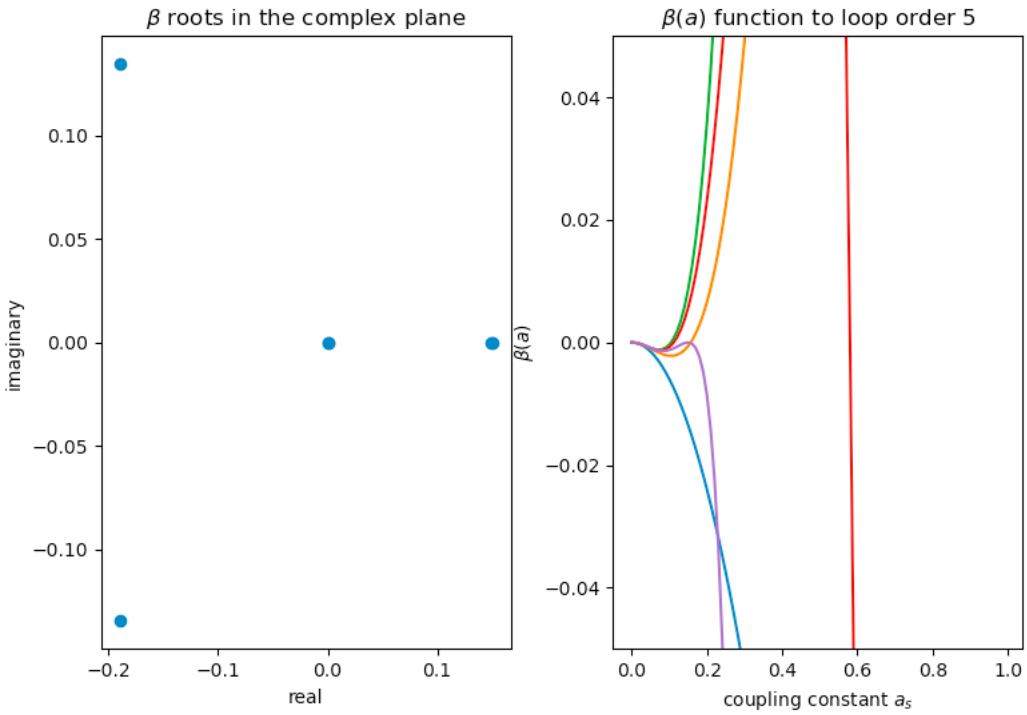


Figure 5: $n_f = 12.89$ β has two real roots: An infrared fixed point with multiplicity 2 at 0.15. Notice the purple loop order 5 dips and rises with the apex of the part with negative concavity tangent at the horizontal axis.

There seems to be a lack of definite infrared fixed points in $n_f < 13$ for loop order 5. Only one fixed point is found in an unrealistic spot between $nf = 12$ and $nf = 13$. Lower order behavior also starts to deviate around the areas of interest. Loop orders 3 and 4 both show UV fixed points from $n_f > 8$. Loop order 5 does not show this behavior. It may be that around values of $a_s > 0.2$ the perturbations for these lower loop orders fail to describe the β function. It is still concerning that these two previous loop orders agree with each other so closely, but largely differ with loop order 5 which is supposed to be a more accurate representation. Values of IR fixed points remain inconclusive as a result and suggest that perhaps a better approximation is warranted. One hopeful method that is still perturbative is the Pade approximate method.

Pade Approximation

The β function can be approximated by way of the Pade approximation. The Pade approximation is similar to a power series except that it assumes the

function is a ratio of two polynomials.

$$F(x) = \frac{P(x)}{Q(x)}$$

$$\sum_{i=0}^{\infty} f_i x^i \approx \frac{\sum_{j=0}^{L} p_j x^j}{\sum_{k=0}^{M} q_k x^k}$$

$$= f_0 + f_1 x^1 + f_2 x^2 + ... + f_n x^n = \frac{p_0 + p_1 x^1 + p_2 x^2 ... p_L x^L}{1 + q_1 x^1 + q_2 x^2 ... q_M x^M}$$

q_0 is typically normalized to 1. For transcendental functions such as the exponential $F(x)$ is taken as the Taylor series expansion.

$$e^x = \sum_{0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{1}{2!}x^2 + ... + \frac{1}{n!}x^n$$

The objects of interest are then the coefficients of the $P(x)$ and $Q(x)$ polynomials. These are gained by multiplying through by $Q(x)$ and setting the coefficients equal to each other.

$$\begin{aligned}(f_0 + f_1 x^1 + f_2 x^2 + ... + f_L x^L)(q_0 + q_1 x^1 + q_2 x^2 + ... + q_M x^M) \\ = p_0 + p_1 x^1 + p_2 x^2 ... p_L x^L \\ \implies f_0 q_0 = p_0 \\ f_0 q_1 + f_1 q_0 = p_1 \\ f_0 q_2 + f_1 q_1 + f_2 q_0 = p_2 \\ f_0 q_M + ... + f_L q_0 = p_L\end{aligned}$$

When the exponent L increases beyond the order of $P(x)$, the coefficients of P become 0 leading to another set of equations. The equations stop when the order (M+L) is reached.

$$\begin{aligned} f_{L-M+1} q_M + ... + f_{L+1} q_0 = 0 \\ \vdots \\ f_L q_M + ... + f_{L+M} q_0 = 0\end{aligned}$$

This system of equations is then solved for the coefficients of $P(x)$ and $Q(x)$. This can easily be thought of as a binomial expansion with a limiting polynomial.

$exp_{2/2}$ can be solved for more clarity. Recall that $q_0 = 1$,

$$\begin{aligned}(1 + x + \frac{1}{2}x^2)(1 + q_1x^1 + q_2x^2) \\ = p_0 + p_1x^1 + p_2x^2 \\ \implies p_0 = f_0 = 1 \\ q_1 + 1 = p_1 \\ q_2 + q_1 + \frac{1}{2} = p_2 \\ q_2 + \frac{1}{2}q_1 = 0 \\ \frac{1}{2}q_2 = 0\end{aligned}$$

Solving this system equations:

$$\begin{aligned}P(x) = 1 + 0.5x + 0.08333x^2 \\ Q(x) = 1 - 0.5x + 0.08333x^2 \\ x = 2 \implies \frac{P}{Q} = 6.99\end{aligned}$$

This is compared with the actual value of $e^2 = 7.39$. More Pade approximants for the exponential function as well as an in depth discussion of the Pade approximate can be found at [5]. Notice that the two values have about 5% fluctuation in there value. The amount of terms in the numerator and denominator vary by L and M respectively. At low L and M, the calculation can be fairly different depending on the choice made. I note this as a potential source of error when working with the β function. The notation for the Pade approximate of function F is typically $F_{L/M}$ or just $[L/M]$.

Data and Pade Analysis

The β function (6) coefficients are approximated by this method in [4]. I chose a slightly different but similar method for the Pade approximate analysis. In (6) the power series starts with a quadratic term. Ryttov and Shrock chose to modify this equation to start with a constant term, however I chose instead to add two extraneous terms to the Beta function for the linear and constant terms with coefficients equal to zero. This was done primarily for ease of implementation with previous programming functionality. I used Python to generate graphs for which the n_f values can be varied using a slider. I did this for the Pade approximate as well as the roots for the original 5 loop β function. Pertinent code that I developed by myself can be located in the Appendix.

For the 5 loop order β function there can only be 3 total Pade approximates because L and M values are constrained by the coefficients b_n where here n is

the loop order plus two additional terms. For clarification:

$$\begin{aligned} \beta_{5l} =& b_0 a_s^2 + b_1 a_s^3 + ... + b_4 a_s^6 \\ \rightarrow \beta_{5l} =& b_0 + b_1 a_s + b_2 a_s^2 + ... + b_6 a_s^6 = \frac{P(a_s)}{Q(a_s)} \\ \therefore & L + M = 6 \end{aligned}$$

I was able to confirm infrared zero points as suggested by table IV in [4], though my L and M numbers do differ from the tabulated ones. I have included it here for easy reference.

TABLE IV: Values of $\alpha_{IR,n\ell,[p,q]}$ from $[p,q]$ Padé approximants to $\beta_{r,5\ell}$, as a function of $N_f \in I_{IRZ}$, including comparison with $\alpha_{IR,4\ell}$ and $\alpha_{IR,5\ell}$. The symbols (i) zp and (ii) pcl mean that the Padé approximant has (i) a coincident zero-pole pair closer to the origin, (ii) a pole or complex-conjugate pair of poles closer to the origin in the complex α plane. Entries with $-$ are unphysical.

N_f	$\alpha_{IR,4\ell}$	$\alpha_{IR,5\ell}$	$\alpha_{IR,5\ell,[3,1]}$	$\alpha_{IR,5\ell,[2,2]}$	$\alpha_{IR,5\ell,[1,3]}$
9	1.072	–	1.02_{zp}	–	–
10	0.815	–	0.756_{zp}	–	pcl
11	0.626	–	0.563_{zp}	–	pcl
12	0.470	–	0.4075_{zp}	0.634	0.614
13	0.337	0.406	–	0.376	0.375
14	0.224	0.233	–	0.232	0.232
15	0.126	0.12 7	0.127	0.127	0.127
16	0.0398	0.0398	0.0398	0.0398	0.0398

In figures 6 and 7 below, we can compare values for $n_f = 12$ and $n_f = 13$ to the Ryttov and Shrock values in table IV. The conversion is $a_s \equiv \alpha/\pi$. $\beta_{2,4}$ values seem to be meaningless due to the existence of poles near zero and their asymptotic behavior. I found values by graphical inspection of $\beta_{4,2}$ and $\beta_{3,3}$ to be $a_s = 0.201$ and $a_s = 0.195$ respectively where $n_f = 12$. Table IV suggests that there are fixed points for $\alpha_{IR,5l,[2,2]}$ at $a_s = 0.202$ where $n_f = 12$ and $a_s = 0.120$ where $n_f = 13$. Also for $\alpha_{IR,5l,[1,3]}$, $a_s = 0.195$ where $n_f = 12$ and $a_s = 0.120$ where $n_f = 13$. This suggests the following correlations between $\alpha_{IR,5l,[L,M]}$ and my $\beta_{L,M}$:

$$\begin{aligned} \alpha_{IR,5l,[3,1]} &= \beta_{2,4} \\ \alpha_{IR,5l,[2,2]} &= \beta_{4,2} \\ \alpha_{IR,5l,[3,1]} &= \beta_{3,3} \end{aligned}$$

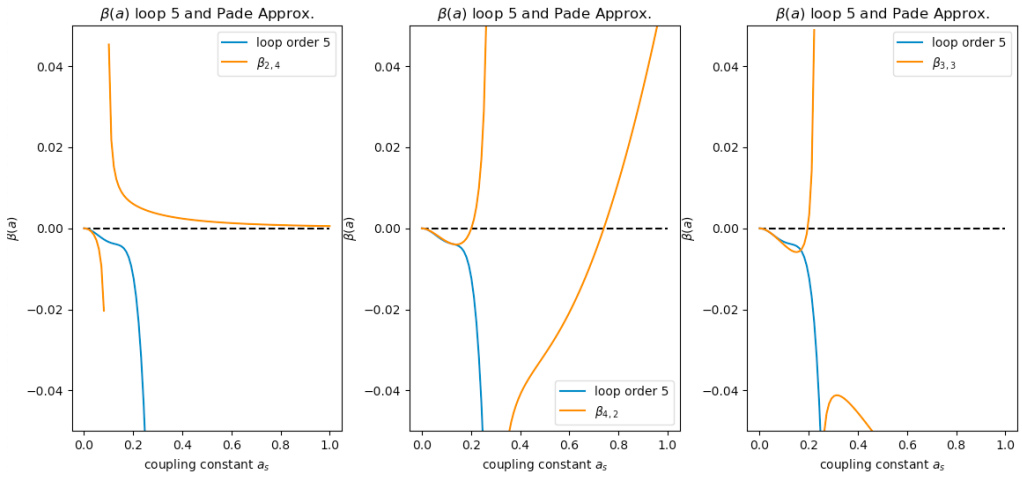


Figure 6: These are the Pade approximate graphs of the 5 loop β for $n_f = 12$. From left to right IR fixed points: $\beta_{2,4} \rightarrow$None, $\beta_{4,2} \rightarrow 0.201$ and $\beta_{3,3} \rightarrow 0.195$.

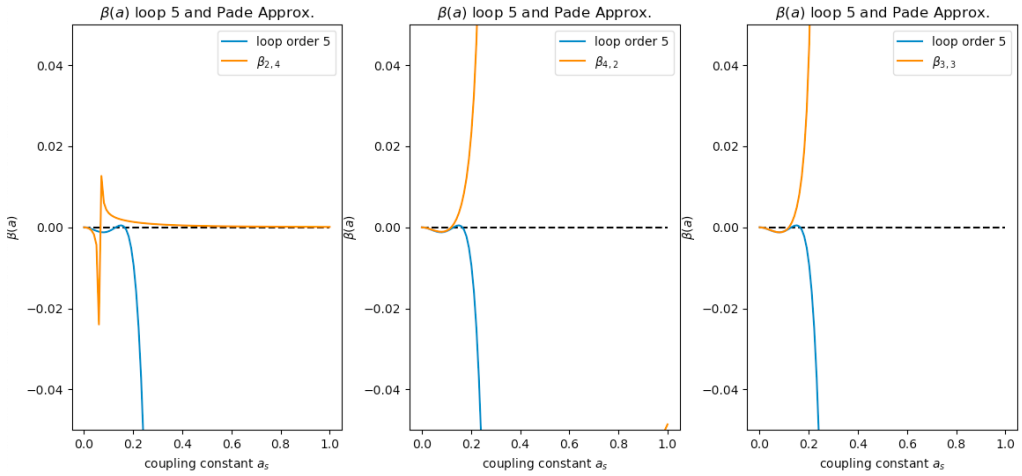


Figure 7: These are the Pade approximate graphs of the 5 loop β for $n_f = 13$. From left to right IR fixed points: $\beta_{2,4} \rightarrow$None, $\beta_{4,2} \rightarrow 0.120$ and $\beta_{3,3} \rightarrow 0.120$. Note that for $\beta_{2,4}$ there is a spurious asymptote connected continuously.

I had hypothesized that the L and M values would line up in a fairly linear fashion with [3,1] corresponding to [4,2], [2,2] to [3,3] and [1,3] to [2,4]. The mismatched indices require further investigation. Stronger graphical resolution might shed light on the discrepancies as well as a reworking of the code found in the appendix. This doesn't seem to affect the results, however, and for the

purposes of this paper I have successfully reproduced the tabulated values. $\beta_{4,2}$ seems to be the be the best approximate scheme for all values of n_f.

While I chose to look at the $n_f = 13$ model throughout, it is only for comparison as it seems that values of $n_f > 13$ consistently have IR fixed points for traditional β function loop orders (> 3) and Pade approximate with full agreement from [4]. Loop orders between 8 and 12 inclusively are not well approximated by Pade due to the poles moving toward the origin. This can be seen clearly in figures 6 and 7 for $\beta_{2,4}$. Where this behavior is not present there seem to be no IR fixed points except for $\beta_{3,3}$ at $a = 0.13$ where $n_f = 8$. This fixed point seems very spurious due to the fact that $n_f = 7$ and $n_f = 9$ both report negative β functions for all L and M values and the approximants are extremely close to the original perturbative 5 loop.

Conclusion

The prospects of gauge theories for an n_f range of 8-12 remain up for debate. For any loop order the range 13-16 seem to unanimously indicate UV fixed points. The 5 loop perturbative approximation for the β function shows that these models remain asymptotically free with no infrared (or ultraviolet) fixed points. This is in controversy to previous loop orders (3 and 4) which predicted IR fixed points. Consecutive analysis of complex roots for the β function so a convergence for a IR fixed point at a value around $n_f = 12.89$. A Pade approximate scheme was used in order to determine if the β function could have a more accurate representation.

The Pade method reveals that there might be an IR fixed point at a value $a_s = 0.20$ for 5 loop $n_f = 12$, however, There are two major concerns for the Pade representation. First, the location of the poles close to the UV fixed point is disconcerting. In fact, the approximate $\beta_{2,4}$ can not be used as a result. This asymptotic behavior for $\beta_{4,2}$ and $\beta_{3,3}$ lead to fast divergences from the original function. Another issue is that varying L and M at low order can vary widely. While it is a consolation that $\beta_{4,2}$ and $\beta_{3,3}$ seem to agree in values for the IR fixed point, this is only an agreement to the first decimal place. Actual values and Pade approximates for conventional known functions, such as the exponential, can not only vary widely but be fairly inaccurate depending on the choices of values for L and M at low orders.

While locations of IR fixed points here remain uncertain, continuing in my own research, I would increase graphical resolution and discern why my Pade approximate relates to Ryttov and Shrock's the way it does. Closer inspection of the trends of lower loop orders as well as the 5 loop might also make for a more full analysis. Unfortunately, a 6th term for the β function will not be calculated for quite a while. A natural progression would be to look at IR fixed points from the perspective of numerical analysis and lattice gauge theory. Another possible approach is to vary the symmetry. All of the models in this paper pertain to the SU(3) symmetry group. Certain symmetry groups might also limit the number of flavors.

References

[1] The Nobel Prize in Physics 2004. 2004. url: https://www.nobelprize.org/prizes/physics/2004/ popular-information/.

[2] The Higgs Boson. 2021. url: https://home.cern/science/physics/higgs-boson

[3] P. A. Baikov, K. G. Chetyrkin and J. H. Kühn, "Five-Loop Running of the QCD coupling constant," Phys. Rev. Lett. **118**, no.8, 082002 (2017) doi:10.1103/PhysRevLett.118.082002 [arXiv:1606.08659 [hep-ph]].

[4] T. A. Ryttov and R. Shrock, "Infrared Zero of β and Value of γ_m for an SU(3) Gauge Theory at the Five-Loop Level," Phys. Rev. D **94**, no.10, 105015 (2016) doi:10.1103/PhysRevD.94.105015 [arXiv:1607.06866 [hep-th]].

[5] Weisstein, Eric W. "Padé Approximant." From MathWorld–A Wolfram Web Resource. https://mathworld.wolfram.com/PadeApproximant.html

[6] Z. Fodor, K. Holland, J. Kuti, D. Nogradi and C. H. Wong, 'Extended investigation of the twelve-flavor β-function," Phys. Lett. B **779**, 230-236 (2018) doi:10.1016/j.physletb.2018.02.008 [arXiv:1710.09262 [hep-lat]].

[7] Griffiths, David J. *Introduction to Electrodynamics*. Fourth Edition. 2013. Pearson Education, Inc.

[8] Thornton, Stephen T. and Rex, Andrew *Modern Physics for Scientists and Engineers*. Fourth Edition. 2013. Brooks/Cole, Cengage Learning.

[9] D. J. Gross and F. Wilczek, Phys.Rev.Lett. 30, 1343 (1973).

[10] H. D. Politzer, Phys.Rev.Lett. 30, 1346 (1973).

Appendix

A.1 Beta telemetry code

This code is designed to vary values of n_f and display all zeros and all loop orders. All coding is done in Python 3.8.3.

```
'''This code plots the Beta function as a function of
    quark-gluon coupling with the number of flavors as a free parameter'''

import numpy as np
from matplotlib.widgets import Button, Slider
import matplotlib.pyplot as plt
import numpy.polynomial.polynomial as poly
from decimal import *

#to get better precision the Decimal library is used here.
getcontext().prec=100

#some initial conditions for the plots
scaleFac=12
fig, ax =plt.subplots(1,3,figsize=[scaleFac,scaleFac*9/16])
fig.tight_layout(pad=3.0)
plt.subplots_adjust(bottom=0.25,wspace=0.25)

#number of flavors
nf0=13
a=np.linspace(0,1,100)
alph0=a[:,np.newaxis]


def draw(y,Roots):
        '''This function is used to draw three graphs: B vs a, complex roots, li
        for i in range(3):
                ax[i].clear()
        ax[2].set_ylim([-0.05,0.05])
        ax[2].set_xlabel(r'coupling constant $a_s$')
        ax[2].set_ylabel(r'$\beta(a)$')
        ax[2].set_title(r'$\beta(a)$ function to loop order 5')
        for i in range(len(y[0,:])):
                ax[2].plot(alph0,-y[:,i],label=f'loop order {i+1}')
        ax[2].legend()

        #plotting zeros
```

```
        ax[1].set_xlabel(r'real')
        ax[1].set_ylabel(r'imaginary')
        ax[1].set_title(r'$\beta$ roots in the complex plane')
        ax[1].scatter(Roots.real,Roots.imag,marker='o')

        #plotting telemetry
        ax[0].axis('off')
        ax[0].set_title('Complex Root Telemetry')
        bRootc=Roots[:,np.newaxis]
        for i in range(len(bRoot)):
                PForm=f'{np.around(bRootc[i].real,decimals=3)} + {np.around(bRoo
                ax[0].text(0.5,0.9-0.1*i,PForm)

#array of zeta values (1 is 0 and not inf)
zeta=np.array([-0.5,0,
               1.644934066848226436472415166646025189218949901206798437735558229
               1.202056903159594285399738161511449990764986292340498881792271555
               1.082323233711138191516003696541167902774750951918726907682976215
               1.036927755143369926331365486457034168057080919501912811974192677

# Two of the beta functions that include zeta functions
def beta(alph, nf):
        '''This function takes in a linspace array and a value for flavors and r
        the roots of the beta function'''
        #coefficients of beta
        b0=(0.25)*(11-2*nf/3)
        b1=(0.25**2)*(102-38*nf/3)
        b2=(0.25**3)*(2857/2-5033*nf/18+325*(nf**2)/54)
        b3=(0.25**4)*(149753/6+3564*zeta[3]-(1078361/162+6508*zeta[3]/27)*nf
                                +(50056/162+6472*zeta[3]/81)*nf**2
                                +(1093/729)*nf**3)
        b4=(0.25**5)*(8157455/16+621885*zeta[3]/2-88209*zeta[4]/2-288090*zeta[5]
                                +(-336460813/1944-4811164*zeta[3]/81+33935*zeta
                                +(25960913/1944+698531*zeta[3]/81-10526*zeta[4]/
                                +(-630559/5832-48722*zeta[3]/243+1618*zeta[4]/27
                                +(1205/2916-152*zeta[3]/81)*nf**4)
        #pre-reduced functions
        betRed3=114.23-27.1339*nf+1.58238*nf**2+0.0058567*nf**3
        betRed4=524.56-181.8*nf+17.16*nf**2-0.22586*nf**3-0.0017933*nf**4

        '''one array has coefficients, another array is made to have consequtive
        polinomial terms (a^2, a^3...etc.), they are then multiplied and each
        column of the resulting array is added to the index below it (this gives
        the recursive relationship b_n=b_(n-1)+b_n*a^(n+2))'''
        bCo=np.array([b0,b1,b2,b3,b4] )
        alphRay=np.full((len(alph),len(bCo)),alph )
```

```
        alphPow=np.multiply.accumulate(alphRay, 1 )
        bTrueCo=alphPow*bCo*alph
        bAcc=np.add.accumulate(bTrueCo,1 )
        '''Here I am inserting 0s in a different array for finding roots'''
        bAt0=np.insert(bCo,[0,0],0)

        return bAcc,poly.polyroots(bAt0)

#plot beta vs a
yval,bRoot=beta(alph0,nf0)
draw(yval,bRoot)


#nf slider
nfAx=plt.axes([0.25, 0.1, 0.65, 0.03], facecolor='lightgrey')
nfSlide = Slider(nfAx, 'Flavor # ', 0, 17, valinit=nf0, valstep=1)
def update_nf(val):
        flav = nfSlide.val
        newY,newRoot=beta(alph0, flav)
        draw(newY,newRoot)
        fig.canvas.draw_idle()
nfSlide.on_changed(update_nf)


plt.show()
```

A.2 Pade Approximate code

During my exploration of the Beta function I was using a lot of the same functionality to research the Pade approximate. The amount of code eventually became unwieldy so that I was forced to rework it to make an object-oriented version. I have here a header file with a Beta object similar to the Beta telemetry code and then a graphing function to graph the Pade against the 5 loop β function. There may be redundancy, but the object should be able to be used to graph the Beta telemetry found in the previous section as well. It may require a bit of improvement or slight modification to graph lower loop orders.

A.2.1 Header File

```
    import numpy as np
import pandas as pd
import numpy.polynomial.polynomial as poly
from matplotlib import pyplot as plt
import tkinter as tk
from matplotlib.widgets import Button, Slider
```

```
'''This is a beta object complete with Pade approx. capability'''
class Beta:
        def __init__(self, nf=12,L=3,M=3,a=0.634/np.pi,a_range=None):
                self.nf=nf
                self.M=M
                self.L=L
                self.a=a
                self.aMax=L+M
                if a_range is None:
                        a_range=np.linspace(0,1,100)
                self.a_range=a_range
                zeta=np.array([-0.5,0,
                                1.64493406684822643647241516664602518921894990
                                1.20205690315959428539973816151144999076498629
                                1.08232323371113819151600369654116790277475097
                                1.03692775514336992633136548645703416805708091
                b0=(0.25)*(11-2*nf/3)
                b1=(0.25**2)*(102-38*nf/3)
                b2=(0.25**3)*(2857/2-5033*nf/18+325*(nf**2)/54)
                b3=(0.25**4)*(149753/6+3564*zeta[3]-(1078361/162+6508*zeta[3]/27
                                        +(50056/162+6472*zeta[3]/81)*nf**2
                                        +(1093/729)*nf**3)
                b4=(0.25**5)*(8157455/16+621885*zeta[3]/2-88209*zeta[4]/2-288090
                                        +(-336460813/1944-4811164*zeta[3]/81+33
                                        +(25960913/1944+698531*zeta[3]/81-10526
                                        +(-630559/5832-48722*zeta[3]/243+1618*z
                                        +(1205/2916-152*zeta[3]/81)*nf**4)

                self.bCo=-1*np.array([0,0,b0,b1,b2,b3,b4])
        def set_nf(self, nf):
                self.nf=nf
        def set_LM(self, L,M):
                self.L=L
                self.M=M
                self.aMax=M+L
        def set_a_range(self, a_Max,a_Min,a_Step):
                self.a_range=np.linspace(a_Max,a_Min,a_Step)
        def set_a(self,a):
                self.a=a
        def pade_solve(self):
                '''This function use a matrix to solve the system of eqs. It ret
                of coefficients for p(x) and q(x)'''
                cofA=pd.DataFrame(np.zeros([self.aMax,self.aMax]))
                for i in range(self.M):
                        cofA.loc[i:, [i]] = self.bCo[:self.aMax-i,np.newaxis]
```

```
            for j in range(self.L):
                    cofA.loc[j, [self.M+j]] = -1
            pqVal=np.linalg.solve(cofA,self.bCo[1:])
            q_cof=pqVal[:self.M]
            p_cof=pqVal[self.M:]
            q_cof=np.insert(q_cof,0,1)
            p_cof=np.insert(p_cof,0,self.bCo[0])
            return p_cof,q_cof
    def pade_val(self):
            '''This function uses the coefficient arrays to compute an actua
            p/q'''
            pcof,qcof=self.pade_solve()
            xpInit=np.full(len(pcof)-1,self.a)
            xqInit=np.full(len(qcof)-1,self.a)
            xPPol=np.multiply.accumulate(xpInit)
            xqPol=np.multiply.accumulate(xqInit)
            p_over_q=(pcof[0]+np.sum(pcof[1:]*xPPol))/(qcof[0]+np.sum(qcof[1
            return p_over_q
    def curve(self,pn=None,x=None):
            '''Here we take polynomial coefficients and add a range of x val
            constant term is added and the input coefficient array is mapped
            if pn is None:
                    pn=self.bCo
            self.pn=pn
            if x is None:
                    x=self.a_range[:,np.newaxis]
            xInit=np.full((len(x),len(pn)-1),x)
            xPol=np.multiply.accumulate(xInit,1)
            padPol=pn[0]+np.sum(pn[1:]*xPol,axis=1)
            return padPol
    def zeros(self,pn=None):
            if pn is None:
                    pn=self.curve()
            self.pn=pn
            return poly.polyroots(self.pn)
```

A.2.2 Pade graph

```
from Beta_Head import *
b=Beta()

#some initial conditions for the plots
scaleFac=12
fig, ax =plt.subplots(1,3,figsize=[scaleFac,scaleFac*9/16])
```

```
fig.tight_layout(pad=3.0)
plt.subplots_adjust(bottom=0.25,wspace=0.25)

def asymp(pade):
        '''This is a small function to eliminate Pade asymptotes connected conti
        the graph. It doesn't work well.''''
        ll=-0.1
        hl=0.1
        pade[pade>hl]=np.inf
        pade[pade<ll]=-np.inf
        return pade
def on_graph(b):
        '''This function is called upon by the slider and graphs the Pade approx
        appropriately'''
        for i in range(3):
                ax[i].clear()
                ax[i].plot(b.a_range, 0*b.a_range,'k--')
                ax[i].set_ylim([-0.05,0.05])
                ax[i].set_xlabel(r'coupling constant $a_s$')
                ax[i].set_ylabel(r'$\beta(a)$')
                ax[i].set_title(r'$\beta(a)$ loop 5 and Pade Approx.')
                ax[i].plot(b.a_range,b.curve(),label=f'loop order 5')
        p,q=b.pade_solve()
        ax[2].plot(b.a_range,asymp(-b.curve(p)/b.curve(q)),label=r'$\beta_{3,3}$
        b.set_LM(4,2)
        p,q=b.pade_solve()
        ax[1].plot(b.a_range,asymp(-b.curve(p)/b.curve(q)),label=r'$\beta_{4,2}$
        b.set_LM(2,4)
        p,q=b.pade_solve()
        ax[0].plot(b.a_range,asymp(-b.curve(p)/b.curve(q)),label=r'$\beta_{2,4}$
        for i in range(3):
                ax[i].legend()

on_graph(b) # initial graph

'''Slider functionality'''
nfAx=plt.axes([0.25, 0.1, 0.65, 0.03], facecolor='lightgrey')
nfSlide = Slider(nfAx, 'Flavor # ', 0, 17, valinit=b.nf, valstep=1)

def update_nf(val):
        flav = nfSlide.val
        beta=Beta(nf=flav)
        on_graph(beta)
        fig.canvas.draw_idle()
nfSlide.on_changed(update_nf)
plt.show()
```