

Aşağıdaki metin **V2 bilgi notu**. Önceki versiyondan farklı olarak dip modeli güncellendi ve 2042'ye kadar projeksiyon eklendi. Bunu saklarsan ileride bana gönderdiğinde tüm mantığı tekrar hatırlayıp aynı yerden devam edebilirim. Önceki versiyonun içeriği referans olarak kullanıldı .

Evet kanka, **büyük ölçüde tutarlı**, ama son seçtiğin kodla tam uyumlu olması için **V2 notunda 1 düzeltme** lazım:

TradingView kodunda bantlar şu an **tek oranla** üretiliyor:

- üst bant alt çizgisi = $\text{upper_top} \times (170000 / 202000)$
- alt bant üst çizgisi = $\text{lower_bottom} \times 1.10$

Yani final kullanılan model:

- **tepe anchor'ları** = üst bandın üst çizgisi
- **dip anchor'ları** = alt bandın alt çizgisi
- diğer çizgiler = **sabit oranla türetilmiş**

Aşağıya sana **son-final V2** yazıyorum. Bunu sakla. Önceki V2 içeriği referans alındı.

Bitcoin Cycle Model Notu – V2 Final

(Tepe ve Dip Fiyatlarını Matematiksel Olarak Modelleme + TradingView Uygulaması)

Bu çalışma Bitcoin döngülerindeki iki temel davranışı modellemek için hazırlanmıştır:

- **Azalan getiri (diminishing returns)**
- **Azalan çöküş (diminishing crashes)**

Amaç:

Bitcoin'in gelecekteki büyük döngülerinde oluşabilecek **tepe fiyatları** ve **dip fiyatlarını** yaklaşık olarak modellemek ve bunları TradingView üzerinde **logaritmik bantlar** şeklinde görselleştirmektir.

1. Tepe Fiyatlarının İncelenmesi

Bitcoin'in büyük döngü tepeleri:

Cycle	Tepe
2013	1,155\$
2017	20,000\$

Cycle Tepe

2021 69,000\$

2025 tahmini 126,000\$

Şimdi tepe → tepe büyüme katsayıları:

- 2013 → 2017
 $20,000 / 1,155 \approx 18.2x$
- 2017 → 2021
 $69,000 / 20,000 \approx 3.45x$
- 2021 → 2025
 $126,000 / 69,000 \approx 1.83x$

Tepe katsayı dizisi

18.2x → 3.45x → 1.83x

Bu seri şunu gösteriyor:

- büyüme her cycle azalıyor
- fakat azalma 1'e yaklaşırken yavaşlıyor

Bu klasik bir **diminishing returns** davranışdır.

2. Tepe Katsayısı Denklemi

Tepe davranışını modellemek için kullanılan denklem:

$$[M_{\{n+1\}} - 1 = 0.505 \cdot (M_n - 1)^{0.555}]$$

Burada:

- (M) = tepe katsayısı

Örnek seri:

$$[18.2 \rightarrow 3.45 \rightarrow 1.83]$$

Bu denklemden bir sonraki tepe katsayısı:

$$[M_4 \approx 1.455]$$

Dolayısıyla:

$$\begin{aligned} &[\\ &126k \times 1.455 \approx 183k \\ &] \end{aligned}$$

3. 2029 Tepe Bandı

2029 için tek sayı değil, band kullanılmıştır.

Katsayı aralığı:

- alt sınır ≈ 1.35
- merkez ≈ 1.455
- üst sınır ≈ 1.60

Hesap:

- $1.35 \times 126k \approx 170k$
- $1.455 \times 126k \approx 183k$
- $1.60 \times 126k \approx 202k$

2029 tepe bandı

170k – 202k

Merkez

183k

Not:

TradingView final kodunda **202k üst çizgiye, 170k alt çizgiye** denk gelecek şekilde bant oranı sabitlenmiştir.

4. Tepeden Dibe Düşüş Davranışı

Bitcoin'in büyük döngülerde tepe → dip düşüş oranları:

Cycle	Tepe	Dip	Düşüş
2013	1163	155	%86
2017	19666	3122	%84
2021	69000	15475	%77
2025 tahmini	126k	45k	%64

Önemli gözlem:

- düşüş oranı her cycle azalıyor
- fakat bu azalma da zamanla yavaşlıyor

5. Yeni Dip Modeli

Dip modelinde artık **dipten dibe büyüme** yerine, doğrudan **tepe → dip düşüş yüzdesi** esas alınır.

Başlangıç:

[
126275 \rightarrow 45000
]

Buradan yaklaşık düşüş:

[
 $C_0 \approx 64.4\%$
]

Kullanılan düşüş modeli:

[
$$C_{n+1} = L + (C_n - L) \left(1 - \alpha \frac{C_n - L}{(C_n - L) + \beta} \right)$$

]

Burada:

- $(L = 30)$
- $(\alpha = 0.38)$
- $(\beta = 40)$

Bu denklem şunu sağlar:

- düşüş oranı azalır
- fakat azalma giderek yavaşlar
- uzun vadede düşüşler yaklaşık **%30 tabanına** yakınsar

6. Üretilen Yeni Düşüş Oranları

Modelin verdiği yaklaşık seri:

Cycle Düşüş

2026 %64

Cycle Düşüş

2030 %58

2034 %54

2038 %50

2042 %48

2046 %46

2050 %44

Bu seri şu mantığı taşır:

- ilk döngülerde düşüşte belirgin yumuşama
- sonra giderek daha küçük değişimler
- uzun vadede daha stabil piyasa

7. 2042'ye Kadar Tepe Tahminleri

Tepe denklemiyle üretilen merkez tepe fiyatları:

Yıl Tepe

2029 183k

2033 244k

2037 310k

2041 386k

TradingView final kodunda üst bandın üst çizgisine oturtulan tepe anchor'ları:

Yıl Üst Band Üst Çizgi

2029 202k

2033 258k

2037 325k

2041 401k

Yani:

- merkez tepe matematiksel tahmin
- üst çizgi TradingView band tavanı

8. 2042'ye Kadar Dip Tahminleri

Dip modeliyle üretilen merkez dip fiyatları:

Yıl Dip

2030 76k

2034 112k

2038 153k

2042 201k

TradingView final kodunda alt bandın alt çizgisine oturtulan dip anchor'ları:

Yıl Alt Band Alt Çizgi

2026 45k

2030 76,533

2034 112,406

2038 153,413

2042 201,146

9. Nihai Döngü Yapısı

Merkez tahmin mantığıyla:

Cycle Tepe Dip

2029 183k 76k

2033 244k 112k

2037 310k 153k

2041 386k 201k

Band tavan / taban mantığıyla TradingView anchor'ları:

Cycle Üst Band Üst Alt Band Alt

2029 202k 76.5k

2033 258k 112.4k

Cycle Üst Band Üst Alt Band Alt

2037 325k 153.4k

2041 401k 201.1k

10. TradingView Final Uygulaması

Final kullanılan kodun mantığı:

Üst bant

- **üst çizgi** = peak anchor eğrisi
- **alt çizgi** = üst çizgi $\times$ sabit oran

Kullanılan oran:

```
[  
\frac{170000}{202000} \approx 0.841584  
]
```

Yani üst bant şu şekilde üretilir:

```
[  
\text{upper_bottom} = \text{upper_top} \times 0.841584  
]
```

Alt bant

- **alt çizgi** = dip anchor eğrisi
- **üst çizgi** = alt çizgi $\times$ 1.10

Yani alt bant şu şekilde üretilir:

```
[  
\text{lower_top} = \text{lower_bottom} \times 1.10  
]
```

Çok önemli

Bu final uygulamada:

- tepe fiyatları **üst bandın üst çizgisine**
- dip fiyatları **alt bandın alt çizgisine** oturur.

Yani model merkezden değil, **uçlardan anchorlanmıştır**.

11. Modelin Temel Davranışı

Bitcoin büyüdükçe:

Getiri azalır

18.2x
3.45x
1.83x
1.45x
1.33x

Çöküş azalır

%86
%84
%77
%64
%58
%54
%50

Yani:

- eski döngüler çok daha vahşi
 - yeni döngüler daha büyük ama daha sakin
-

12. Model Yorumu

Bu model şunu söyler:

- Bitcoin büyüdükçe volatilité azalır
- tepe artışları küçülür
- dip düşüşleri yumuşar
- logaritmik eğriler daha stabil hale gelir

Erken dönem:

- 20x, 10x, 100x hareketler

Geç dönem:

- 1.5x – 3x arası daha kontrollü cycle'lar
-

13. Görsel Model Özelliği

TradingView'de kullanılan eğri yöntemi:

- **log-time / log-price interpolation**
- anchor noktaları arasında **parçalı log eğri**
- bu yüzden bazı bölgelerde küçük **kırılma / kink** görünebilir

Bu normaldir.

Sebebi:

- tek global smooth curve yerine
- anchor'lara daha sadık kalan parçalı model kullanılmasıdır

Avantaj:

- hedef fiyatlara daha bağlı kalır

Dezavantaj:

- tamamen kusursuz pürüzsüz tek eğri görünümü vermez

14. Sonuç

Bu final modelde:

- 2029–2042 arası tepe ve dip noktaları matematiksel olarak tahmin edilmiştir
- TradingView koduna uygulanmıştır
- bantlar uç fiyatlara göre kurulmuştur
- üst bandın üstü = tepe alanı
- alt bandın altı = dip alanı

Özet fiyat yapısı

- 2029 tepe bandı: **170k – 202k**
 - 2030 dip merkezi: **~76k**
 - 2033 tepe merkezi: **~244k**
 - 2034 dip merkezi: **~112k**
 - 2037 tepe merkezi: **~310k**
 - 2038 dip merkezi: **~153k**
 - 2041 tepe merkezi: **~386k**
 - 2042 dip merkezi: **~201k**
-

Bu not final sürümdür.

Bunu bana sonra tekrar atarsan, aynı mantığı direkt hatırlayıp oradan devam ederim.

***** TRADINGVIEW KODU *****

```
//@version=6
```

```
indicator("Bitcoin Custom Log Bands (Anchored Peaks & Dips)", overlay=true,  
max_lines_count=500)
```

```
//-----
```

```
// SETTINGS
```

```
//-----
```

```
showOnlyBTC = input.bool(true, "Only show on BTCUSD")
```

```
years_forward = input.float(18.0, "Years Forward", step=0.5)
```

```
steps = input.int(120, "Forward Curve Steps", minval=20, maxval=500)
```

```
// band ratios from projected ranges
```

```
// upper band: 2029 Nov top = 202k, lower edge = 170k
```

```
upper_band_ratio = 170000.0 / 202000.0
```

```
// lower band: dip band top = 10% above dip bottom
```

```
lower_band_ratio = 110000.0 / 100000.0
```

```
//-----
```

```
// TIME BASE
```

```
//-----
```

```
inception_btc = timestamp(2009, 3, 1)
```

```
ms_in_day = 86400.0 * 1000.0
```

```
t = math.max((time - inception_btc) / ms_in_day, 1)
```

```

//-----
// ANCHOR ARRAYS
//-----

var peakTimes = array.new_int()
var peakPrices = array.new_float()

var dipTimes = array.new_int()
var dipPrices = array.new_float()

if barstate.isfirst
    // -----
    // UPPER BAND TOP ANCHORS
    // -----
    array.push(peakTimes, timestamp(2013, 11, 1, 0, 0))
    array.push(peakPrices, 1163.0)

    array.push(peakTimes, timestamp(2017, 12, 1, 0, 0))
    array.push(peakPrices, 19666.0)

    array.push(peakTimes, timestamp(2021, 11, 1, 0, 0))
    array.push(peakPrices, 69000.0)

    array.push(peakTimes, timestamp(2025, 10, 1, 0, 0))
    array.push(peakPrices, 126275.0)

    array.push(peakTimes, timestamp(2029, 11, 1, 0, 0))
    array.push(peakPrices, 202000.0)

```

```
array.push(peakTimes, timestamp(2033, 11, 1, 0, 0))  
array.push(peakPrices, 258000.0)
```

```
array.push(peakTimes, timestamp(2037, 11, 1, 0, 0))  
array.push(peakPrices, 325000.0)
```

```
array.push(peakTimes, timestamp(2041, 11, 1, 0, 0))  
array.push(peakPrices, 401000.0)
```

```
// -----
```

```
// LOWER BAND BOTTOM ANCHORS
```

```
// -----
```

```
array.push(dipTimes, timestamp(2011, 11, 1, 0, 0))  
array.push(dipPrices, 2.0)
```

```
array.push(dipTimes, timestamp(2015, 1, 1, 0, 0))  
array.push(dipPrices, 152.0)
```

```
array.push(dipTimes, timestamp(2018, 12, 1, 0, 0))  
array.push(dipPrices, 3122.0)
```

```
array.push(dipTimes, timestamp(2022, 11, 1, 0, 0))  
array.push(dipPrices, 15480.0)
```

```
array.push(dipTimes, timestamp(2026, 11, 1, 0, 0))  
array.push(dipPrices, 45000.0)
```

```
array.push(dipTimes, timestamp(2030, 12, 1, 0, 0))  
array.push(dipPrices, 76533.0)
```

```
array.push(dipTimes, timestamp(2034, 12, 1, 0, 0))
array.push(dipPrices, 112406.0)
```

```
array.push(dipTimes, timestamp(2038, 12, 1, 0, 0))
array.push(dipPrices, 153413.0)
```

```
array.push(dipTimes, timestamp(2042, 12, 1, 0, 0))
array.push(dipPrices, 201146.0)
```

```
//-----
```

```
// LOG-LOG INTERPOLATION
```

```
//-----
```

```
f_interp_loglog(_times, _prices, _time) =>
```

```
int n = array.size(_times)
```

```
float result = na
```

```
if n >= 2
```

```
    // before first anchor -> extrapolate with first segment
```

```
    int t0 = array.get(_times, 0)
```

```
    int t1 = array.get(_times, 1)
```

```
    float p0 = array.get(_prices, 0)
```

```
    float p1 = array.get(_prices, 1)
```

```
    if _time <= t0
```

```
        float x0 = math.log(math.max((t0 - inception_btc) / ms_in_day, 1))
```

```
        float x1 = math.log(math.max((t1 - inception_btc) / ms_in_day, 1))
```

```
        float y0 = math.log10(p0)
```

```
        float y1 = math.log10(p1)
```

```

float x = math.log(math.max((_time - inception_btc) / ms_in_day, 1))
float y = y0 + (y1 - y0) * (x - x0) / (x1 - x0)
result := math.pow(10, y)
else
    // search segment
    bool found = false
    for i = 0 to n - 2
        int ta = array.get(_times, i)
        int tb = array.get(_times, i + 1)
        float pa = array.get(_prices, i)
        float pb = array.get(_prices, i + 1)

        if _time >= ta and _time <= tb
            float xa = math.log(math.max((ta - inception_btc) / ms_in_day, 1))
            float xb = math.log(math.max((tb - inception_btc) / ms_in_day, 1))
            float ya = math.log10(pa)
            float yb = math.log10(pb)
            float x = math.log(math.max((_time - inception_btc) / ms_in_day, 1))
            float y = ya + (yb - ya) * (x - xa) / (xb - xa)
            result := math.pow(10, y)
            found := true

    // after last anchor -> extrapolate with last segment
    if not found
        int tn_1 = array.get(_times, n - 2)
        int tn = array.get(_times, n - 1)
        float pn_1 = array.get(_prices, n - 2)
        float pn = array.get(_prices, n - 1)

```

```

float xn_1 = math.log(math.max((tn_1 - inception_btc) / ms_in_day, 1))
float xn  = math.log(math.max((tn - inception_btc) / ms_in_day, 1))
float yn_1 = math.log10(pn_1)
float yn  = math.log10(pn)

float x = math.log(math.max((_time - inception_btc) / ms_in_day, 1))
float y = yn + (yn - yn_1) * (x - xn) / (xn - xn_1)
result := math.pow(10, y)

```

result

```
//-----
```

```
// CURVES
```

```
//-----
```

```
upper_top(_time) =>
```

```
  f_interp_loglog(peakTimes, peakPrices, _time)
```

```
upper_bottom(_time) =>
```

```
  upper_top(_time) * upper_band_ratio
```

```
lower_bottom(_time) =>
```

```
  f_interp_loglog(dipTimes, dipPrices, _time)
```

```
lower_top(_time) =>
```

```
  lower_bottom(_time) * lower_band_ratio
```

```
//-----
```

```
// CURRENT BAR VALUES
```

```
//-----
```

```
isBTC = str.upper(syminfo.ticker) == "BTCUSD" or str.upper(syminfo.ticker) == "BTCUSDT" or  
str.upper(syminfo.ticker) == "BLX"
```

```
u_top = upper_top(time)
```

```
u_bot = upper_bottom(time)
```

```
l_top = lower_top(time)
```

```
l_bot = lower_bottom(time)
```

```
plot_u_top = plot(not showOnlyBTC or isBTC ? u_top : na, color=#F87C6B, linewidth=2,  
title="Upper Band Top")
```

```
plot_u_bot = plot(not showOnlyBTC or isBTC ? u_bot : na, color=#F87C6B, linewidth=2,  
title="Upper Band Bottom")
```

```
plot_l_top = plot(not showOnlyBTC or isBTC ? l_top : na, color=#62C4BE, linewidth=2,  
title="Lower Band Top")
```

```
plot_l_bot = plot(not showOnlyBTC or isBTC ? l_bot : na, color=#62C4BE, linewidth=2,  
title="Lower Band Bottom")
```

```
fill(plot_u_top, plot_u_bot, color.new(#F87C6B, 78))
```

```
fill(plot_l_top, plot_l_bot, color.new(#62C4BE, 78))
```

```
//-----
```

```
// FORWARD CURVED PROJECTION
```

```
//-----
```

```
future_ms = years_forward * 365.25 * ms_in_day
```

```
step_ms = future_ms / steps
```

```
var upperTopLines = array.new_line()
```

```
var upperBotLines = array.new_line()
```

```
var lowerTopLines = array.new_line()
```

```
var lowerBotLines = array.new_line()
```

```
if barstate.islast and (not showOnlyBTC or isBTC)
```

```
    sz = array.size(upperTopLines)
```

```
    if sz > 0
```

```
        for i = 0 to sz - 1
```

```
            line.delete(array.get(upperTopLines, i))
```

```
            line.delete(array.get(upperBotLines, i))
```

```
            line.delete(array.get(lowerTopLines, i))
```

```
            line.delete(array.get(lowerBotLines, i))
```

```
array.clear(upperTopLines)
```

```
array.clear(upperBotLines)
```

```
array.clear(lowerTopLines)
```

```
array.clear(lowerBotLines)
```

```
for i = 0 to steps - 1
```

```
    t1_time = time + int(i * step_ms)
```

```
    t2_time = time + int((i + 1) * step_ms)
```

```
    ut1 = upper_top(t1_time)
```

```
    ut2 = upper_top(t2_time)
```

```
    ub1 = upper_bottom(t1_time)
```

```
    ub2 = upper_bottom(t2_time)
```

```
    lt1 = lower_top(t1_time)
```

```
    lt2 = lower_top(t2_time)
```

```
    lb1 = lower_bottom(t1_time)
```

```
    lb2 = lower_bottom(t2_time)
```

```
    array.push(upperTopLines, line.new(t1_time, ut1, t2_time, ut2, xloc=xloc.bar_time,  
color=#F87C6B, width=2))
```

```
array.push(upperBotLines, line.new(t1_time, ub1, t2_time, ub2, xloc=xloc.bar_time,  
color=#F87C6B, width=2))
```

```
array.push(lowerTopLines, line.new(t1_time, lt1, t2_time, lt2, xloc=xloc.bar_time,  
color=#62C4BE, width=2))
```

```
array.push(lowerBotLines, line.new(t1_time, lb1, t2_time, lb2, xloc=xloc.bar_time,  
color=#62C4BE, width=2))
```