```
// This work is licensed under a Attribution-
NonCommercial-ShareAlike 4.0 International (CC BY-NC-SA
4.0) https://creativecommons.org/licenses/by-nc-sa/4.0/
// © LuxAlgo

//@version=4
study("The Echo Forecast [LUX]","ECHO
[LuxAlgo]",overlay=true,max_bars_back=1000,max_lines_coun
t=200)
length = input(50,'Evaluation
Window',minval=0,maxval=200)
fcast  = input(50,'Forecast Window',minval=1,maxval=200)

fmode  = input('Similarity','Forecast
Mode',options=['Similarity','Dissimilarity'])
cmode  = input('Cumulative','Forecast
Construction',options=['Cumulative','Mean','Linreg'])
src    = input(close)

fcast_col   = input(#2157f3,'Forecast
Style',inline='fcast_style',group='Style')
fcast_style = input('· · ·','',options=['——','– – –','· ·
·'],inline='fcast_style',group='Style')

show_area   = input(true,'Show
Area',inline='areas',group='Style')
fcast_area  =
input(color.new(#ff5d00,50),'',inline='areas',group='Styl
e')
corr_area   =
input(color.new(#0cb51a,50),'',inline='areas',group='Styl
e')
eval_area   =
input(color.new(color.gray,50),'',inline='areas',group='S
tyle')
//----
var lines = array.new_line(0)
if barstate.isfirst
    for i = 0 to fcast-1
```

```
        array.push(lines,line.new(na,na,na,na))
//----
n = bar_index
d = change(src)

top = highest(src,length+fcast*2)
btm = lowest(src,length+fcast*2)

if barstate.islast
    float val = na
    k = 0
    A = array.new_float(0)
    X = array.new_int(0)
    for i = 0 to fcast*2+length
        array.push(A,src[i])
        if cmode == 'Linreg'
            array.push(X,n[i])

    a = array.slice(A,0,fcast-1)
    for i = 0 to length-1
        b = array.slice(A,fcast+i,fcast*2+i-1)
        r = array.covariance(a,b)/
(array.stdev(a)*array.stdev(b))
        if fmode == 'Similarity'
            val := r >= nz(val,r) ? r : val
        else
            val := r <= nz(val,r) ? r : val
        k := val == r ? i : k

    prev = src
    current = src

    for i = 0 to fcast-1
        e = d[fcast+k+(fcast-i-1)]
        if cmode == 'Mean'
            current := array.avg(a) + e
        else if cmode == 'Linreg'
            a = array.slice(A,0,fcast)
            x = array.slice(X,0,fcast)
```

```
            alpha = array.covariance(a,x)/
array.variance(x)
            beta = array.avg(a) - alpha*array.avg(x)
            current := alpha*(n+i+1) + beta + e
        else
            current += e

        l = array.get(lines,i)
        line.set_xy1(l,n+i,prev)
        line.set_xy2(l,n+i+1,current)
        line.set_color(l,fcast_col)

        if fcast_style == '- - -'
            line.set_style(l,line.style_dashed)
        else if fcast_style == '. . .'
            line.set_style(l,line.style_dotted)

        prev := current

    if show_area
        box.delete(box.new(n-length-fcast*2+1,top,n-
fcast+1,btm,
          border_color=na,
          bgcolor=eval_area)[1])
        box.delete(box.new(n-fcast+1,top,n,btm,
          border_color=na,
          bgcolor=fcast_area)[1])
        box.delete(box.new(n-k-fcast*2+1,btm,n-k-
fcast,top,
          border_color=na,
          bgcolor=corr_area)[1])
```