Supplemental material

Daniele Zago[a,*] and Giovanna Capizzi[b]
[a] Department of Statistical Sciences, University of Padua, Italy
https://orcid.org/0000-0003-0778-7099
[b] Department of Statistical Sciences, University of Padua, Italy
http://orcid.org/0000-0002-3187-1365
[*] Corresponding author: daniele.zago.1@phd.unipd.it

1 Further simulations

In the following, additional simulation results are presented for other parameter settings suggested by Aly et al. (2021), namely $(\theta, \lambda) = (4, 0.023)$ and $(7, 0.019)$.

For both settings, Figure 1 and Figure 2 display the in-control CARLs and the distribution of the control limits of the one-sided EWMA control statistic satisfying the GICP condition (Equation (25) in the paper). Figure 3 and Figure 4 show the distribution of the out-of-control CARLs, whereas Figure 5 and Figure 6 show the profiles of the median of the out-of-control CARLs. Analogously to the corresponding figures in the paper, for these profiles the ARLs for $\delta = 0$ are not displayed as they are not directly comparable across different parameter update rules.

(a) $\theta = 4, \lambda = 0.023$ (b) $\theta = 7, \lambda = 0.019$

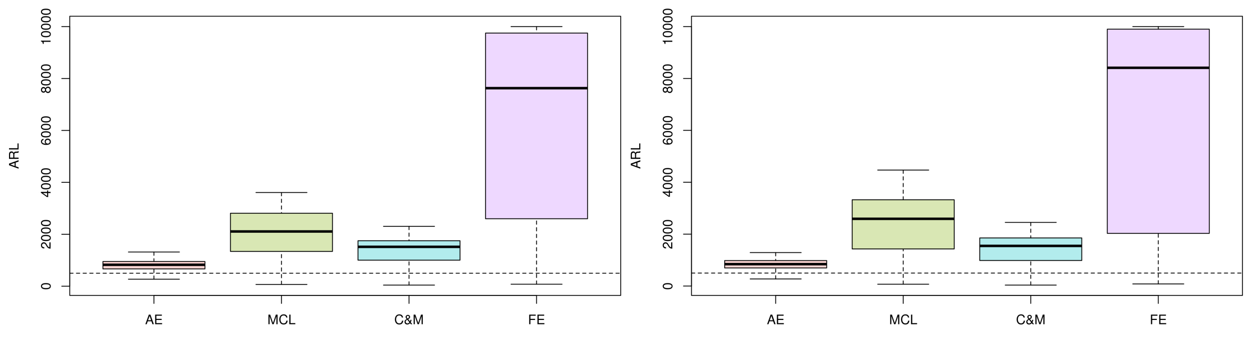
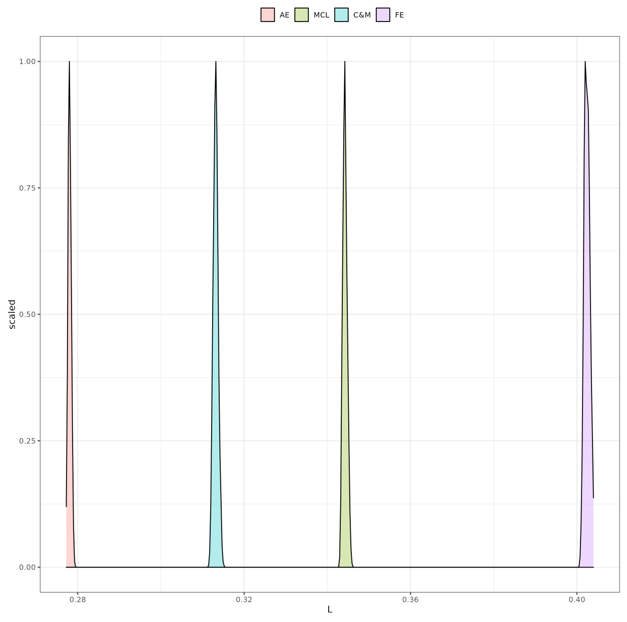


Figure 1: IC performance of the one-sided EWMA control chart under the fixed (FE), adaptive (AE), and two cautious learning parameter schemes. For all the estimation approaches, the corresponding control schemes satisfy the GICP condition ($\beta = 0.1$)

(a) $\theta = 4, \lambda = 0.023$



(b) $\theta = 7, \lambda = 0.019$

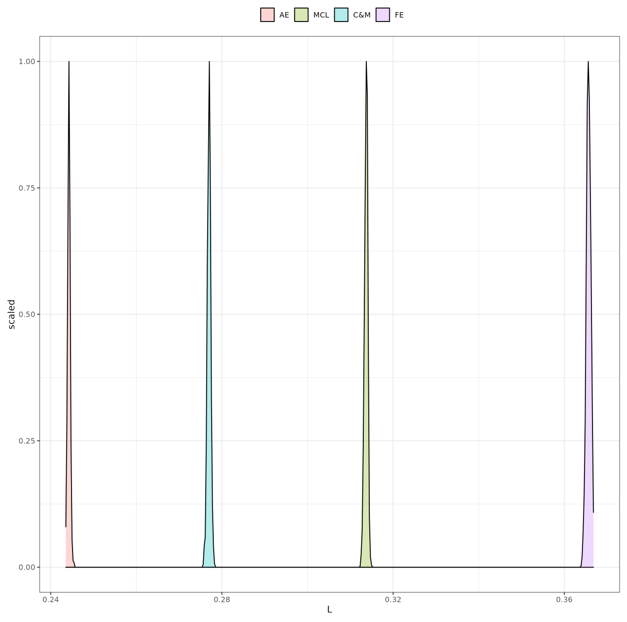


Figure 2: Distribution of the control limits of the EWMA control chart that satisfy the GICP condition ($\beta = 0.1$) using the fixed (FE), adaptive (AE) procedures and two cautious learning parameter schemes. Densities are based on the 200 simulated initial samples.

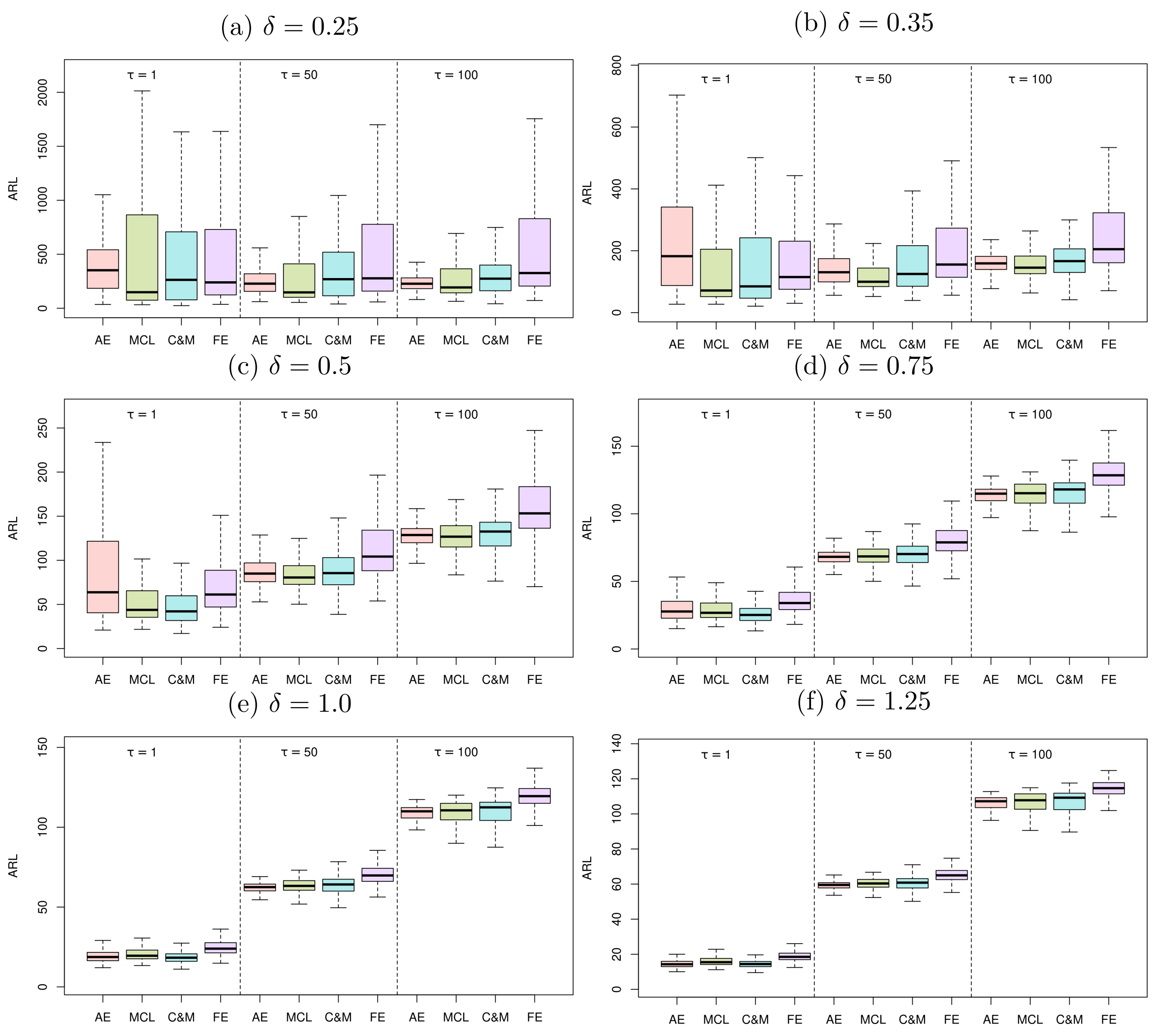


Figure 3: OC performance of the one-sided EWMA control chart based on the fixed (FE), adaptive (AE) procedures and two cautious learning parameter schemes, when $\theta = 4$ and $\lambda = 0.023$.

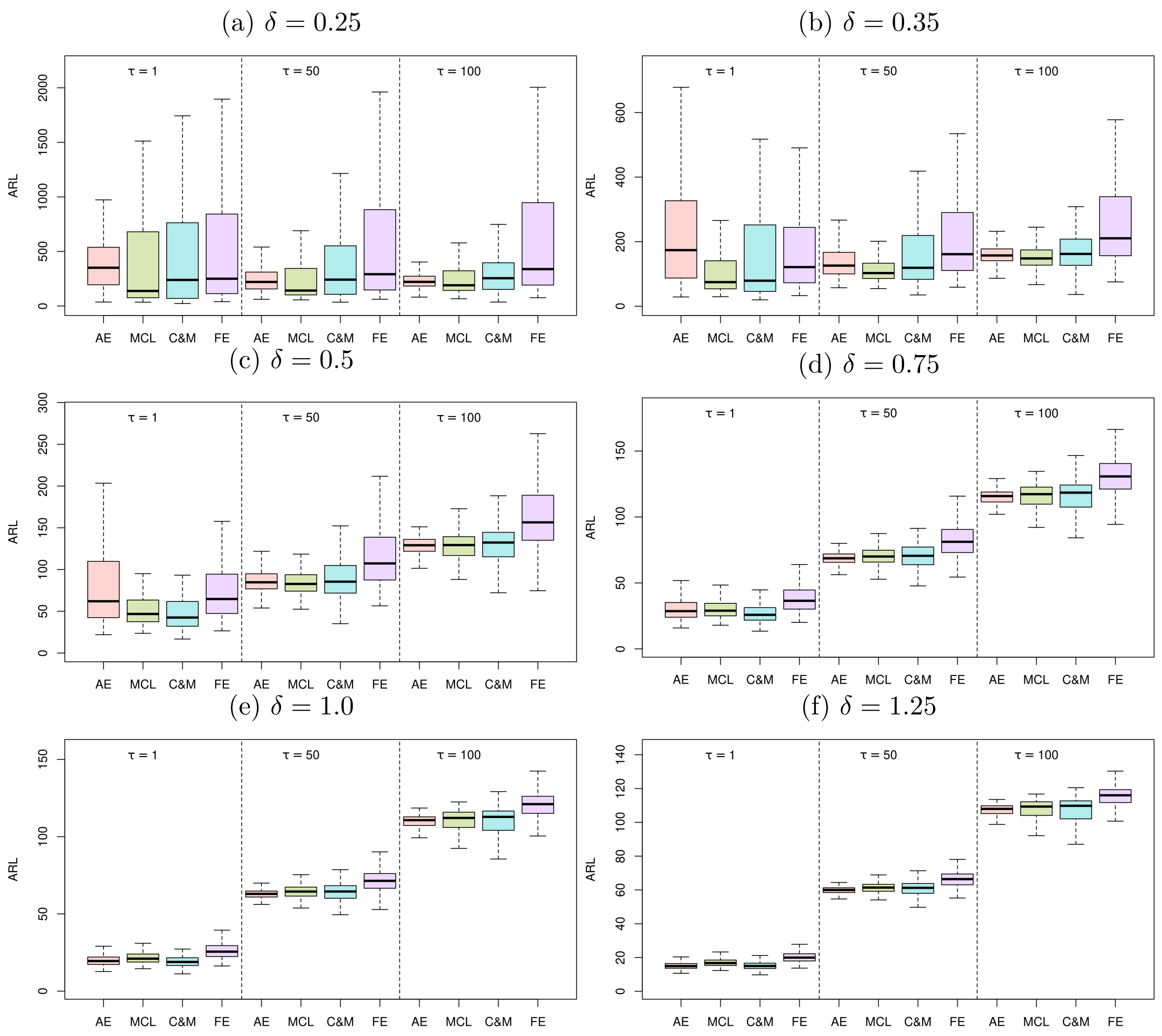


Figure 4: OC performance of the one-sided EWMA control chart based on the fixed (FE), adaptive (AE) procedures and two cautious learning parameter schemes, when $\theta = 7$ and $\lambda = 0.019$.

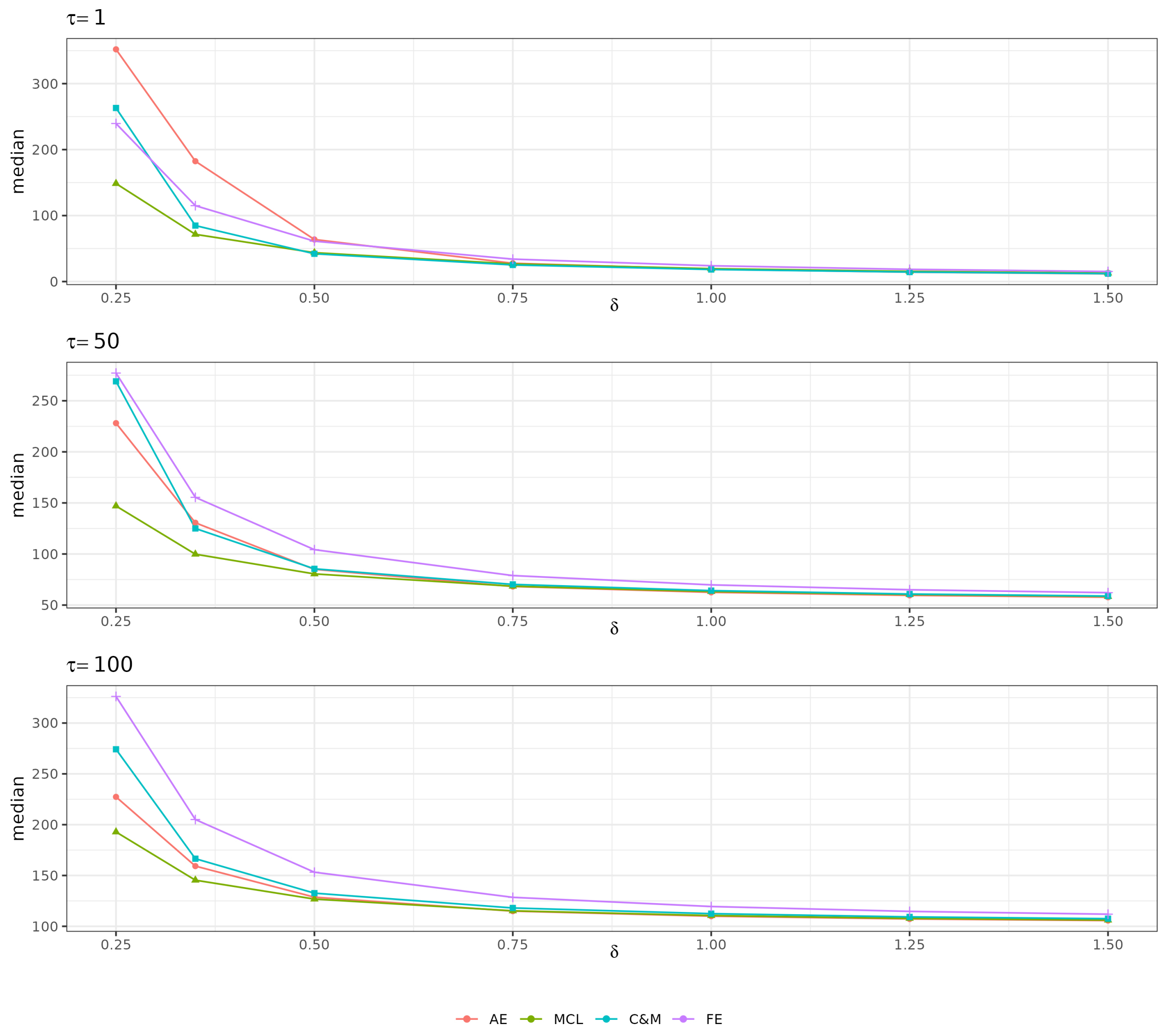


Figure 5: Median of the OC conditional ARL of the one-sided EWMA-type control based on the fixed (FE), adaptive (AE) procedures and two cautious learning parameter schemes, when $\theta = 4$ and $\lambda = 0.023$.

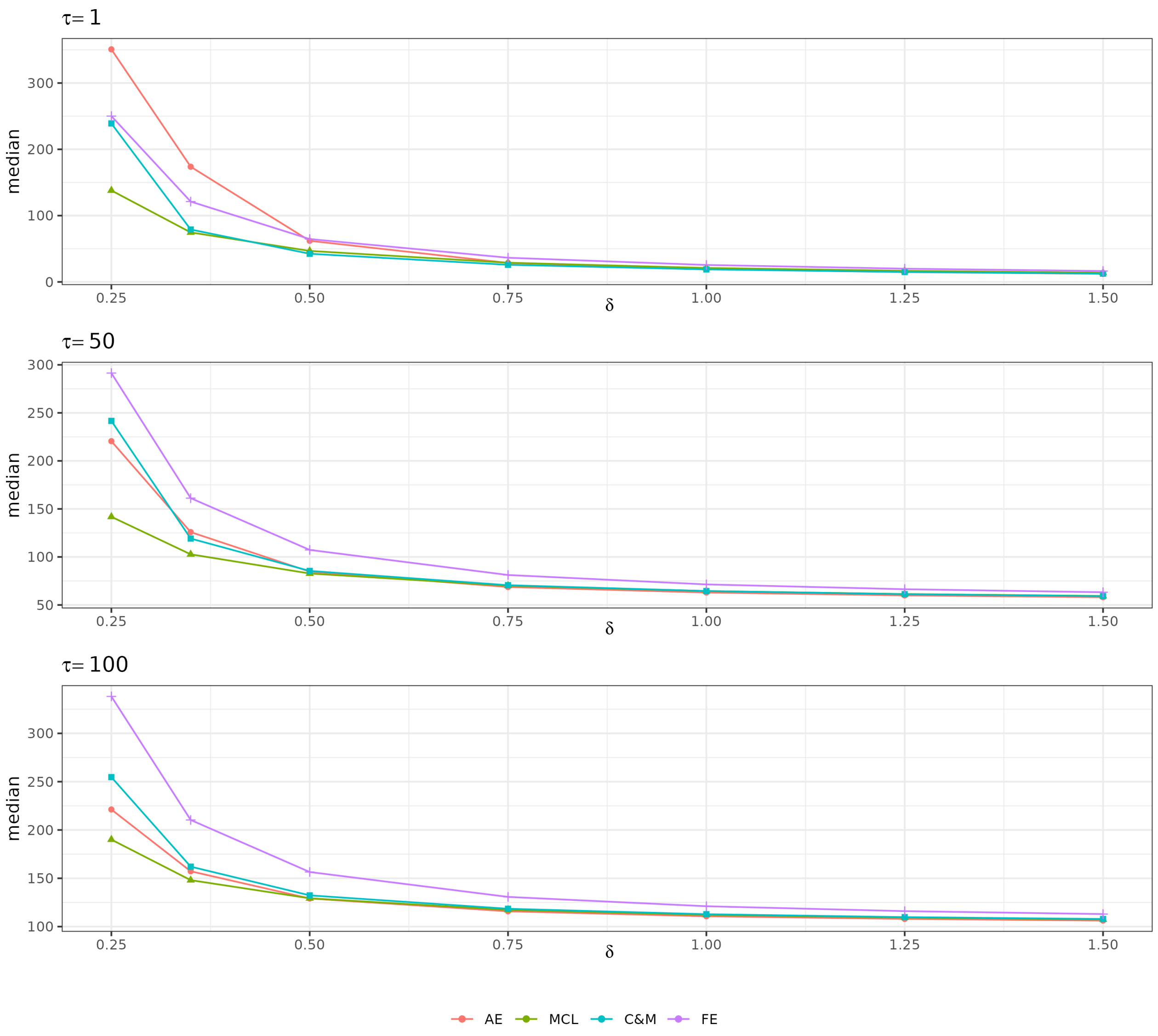


Figure 6: Median of the OC conditional ARL of the one-sided EWMA control based on the fixed (FE), adaptive (AE) procedures and two cautious learning parameter schemes, when $\theta = 7$ and $\lambda = 0.019$.

2 In-depth analysis

In this section a sensitivity analysis of the performance of the MCL and C&M cautious parameter learning schemes is implemented for different values of the smoothing constant λ.

2.1 Simulation settings and results

In the following, the simulation settings are the same as those illustrated in Section 4 of the main paper (same initial sample size, cautious learning parameters, shift magnitudes, change-point locations, and GICP condition for commutating the control limit). The IC process parameter is here set equal to to $\theta = 4$, representing an intermediate value between an heavily skewed and a more symmetric value of $\theta = 1$ and 7, respectively. The performances of the proposed cautious parameter learning schemes rules are compared with those obtained with the FE and AE procedures for the following values of $\lambda = 0.05, 0.075, 0.1, 0.125, 0.15, 0.175$, and 0.2.

From results in Figures 7 to 13, it is possible to see that the time-delayed update rules provide consistently better results with respect those obtained with both adaptive and fixed procedures. This result is stable across different shift sizes and λ values, with some differences between the MCL and the C&M approaches.

Overall, the recommended settings of $A = 1.5$ and $B = 50$ for the C&M parameter learning scheme results in a more conservative rule than that used in the MCL scheme. Indeed, when the control chart displays a high sensitivity to small shifts such as in Figure 7, the MCL approach provides a protection against early shifts comparable to that reachable with the C&M update rule. However, in these cases the MCL approach provides a better protection against shifts occurring later in the monitoring phase due to the higher precision of the estimates.

When the control chart is less sensitive to small parameter shifts, such as for $\lambda = 0.2$ in Figure 13, the MCL approach is not able enough to prevent that small shifts add some bias in parameter estimates. In these cases the C&M approach is more conservative and avoids contamination of the parameter estimates.

Overall, among the proposed time-delayed estimators, it is not possible to find an uniformly outperforming parameter learning scheme. The sensitivity of the control chart based either on the MCL or C&M proposal seems to be affected differently by the size of the mean shifts. However, both the time-delayed estimators lead to a better overall performance than that obtained with FE and AE procedures.

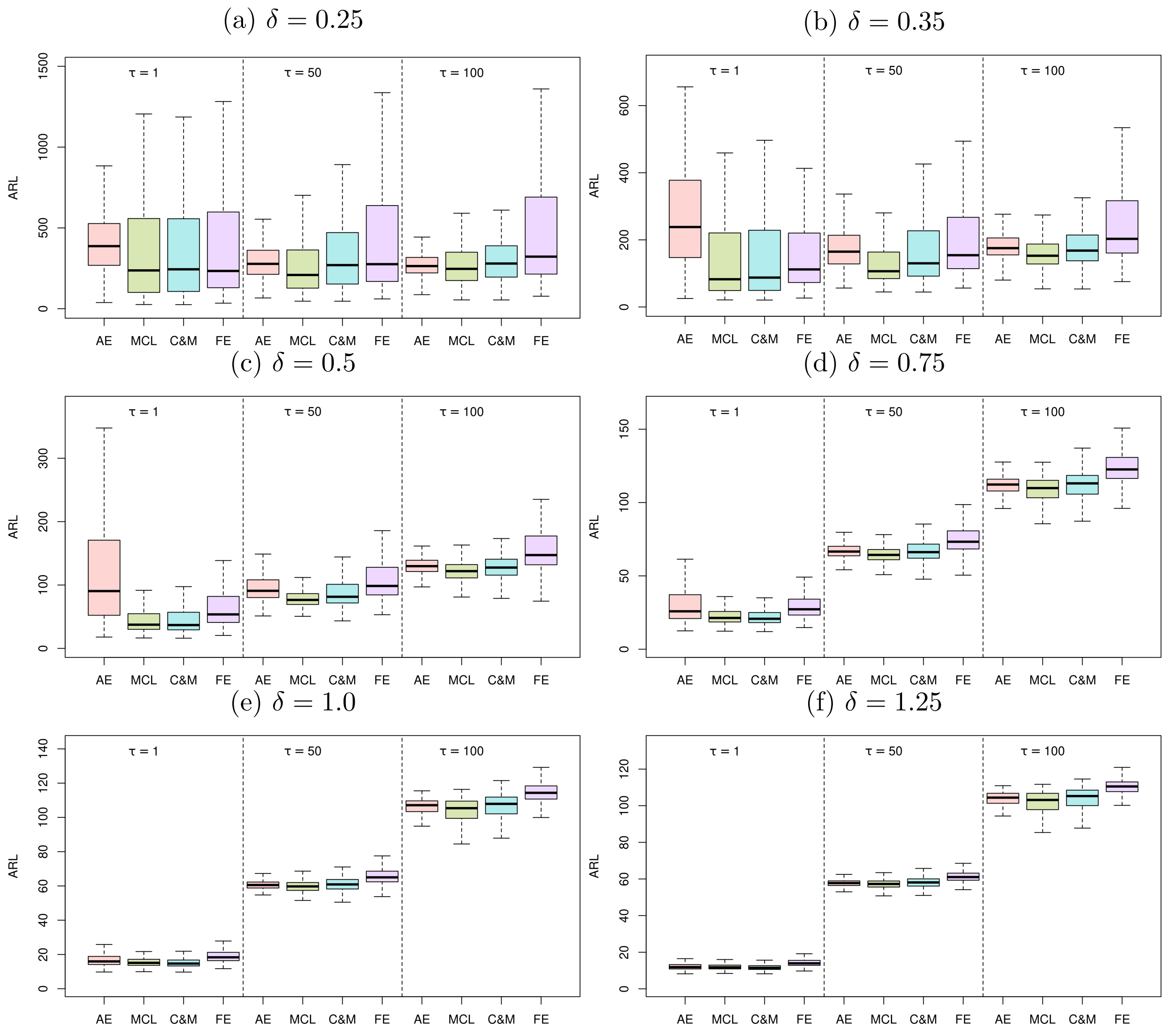


Figure 7: OC performance of the EWMA ($\lambda = 0.05$) control chart based on the fixed (FE), adaptive (AE) procedures, and two cautious learning parameter schemes, when $\theta = 4$. Control charts satisfy the GICP condition with $\beta = 0.1$. Boxplots are based on the 200 simulated conditional ARLs.

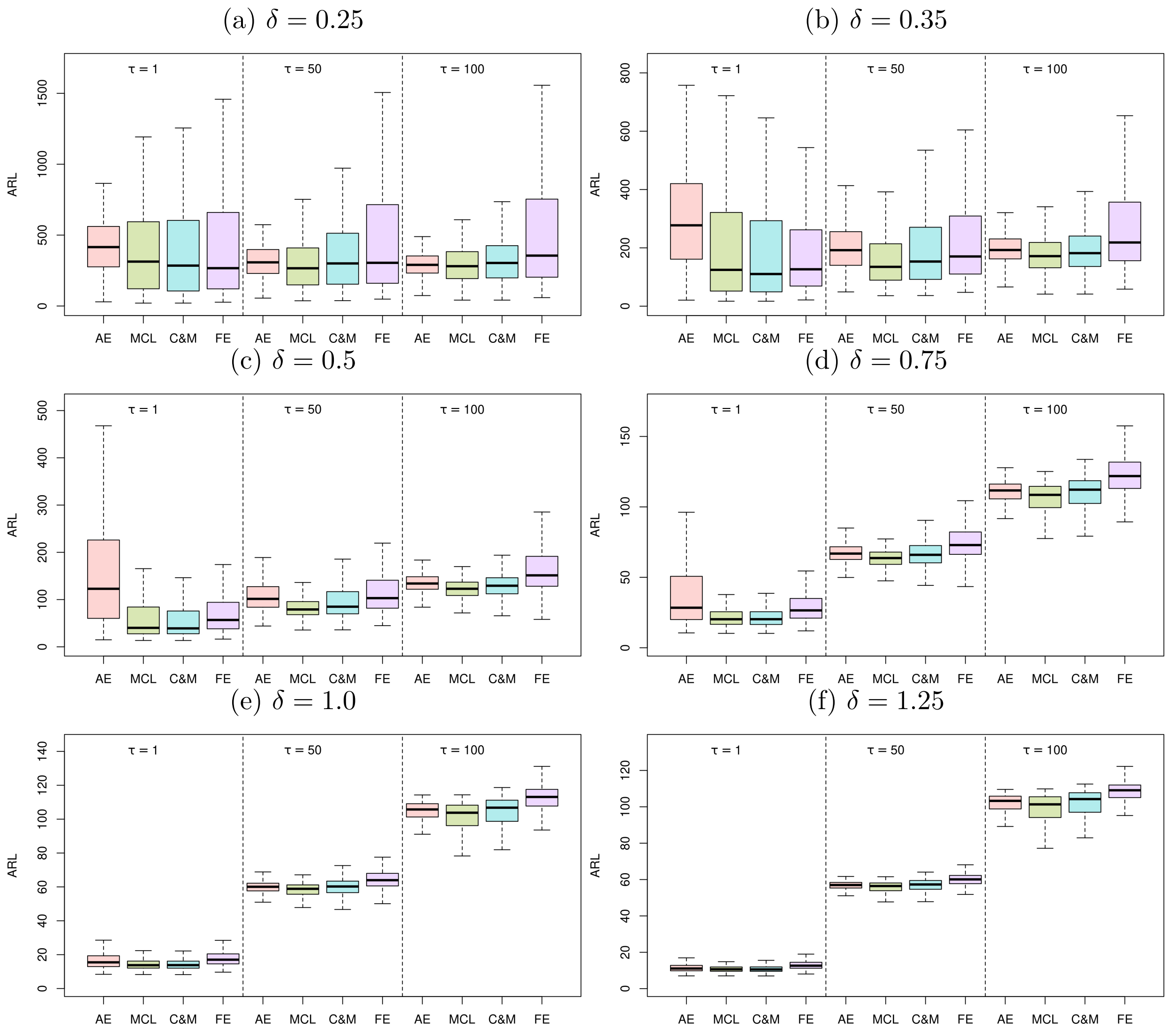


Figure 8: OC performance of the EWMA ($\lambda = 0.075$) control chart based on the fixed (FE), adaptive (AE) procedures, and two cautious learning parameter schemes, when $\theta = 4$. Control charts satisfy the GICP condition with $\beta = 0.1$. Boxplots are based on the 200 simulated conditional ARLs.

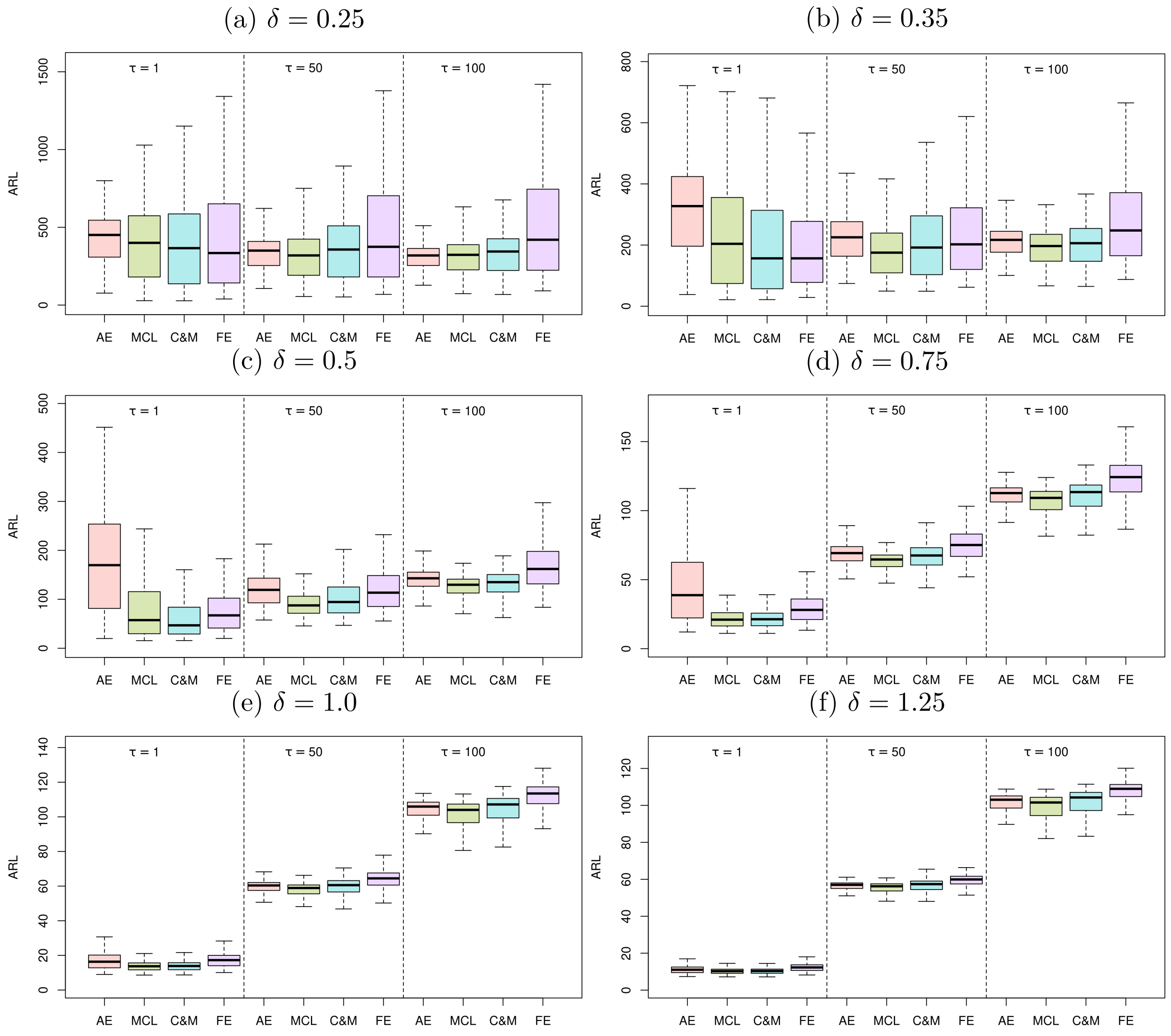


Figure 9: OC performance of the EWMA ($\lambda = 0.1$) control chart based on the fixed (FE), adaptive (AE) procedures, and two cautious learning parameter schemes, when $\theta = 4$. Control charts satisfy the GICP condition with $\beta = 0.1$. Boxplots are based on the 200 simulated conditional ARLs.

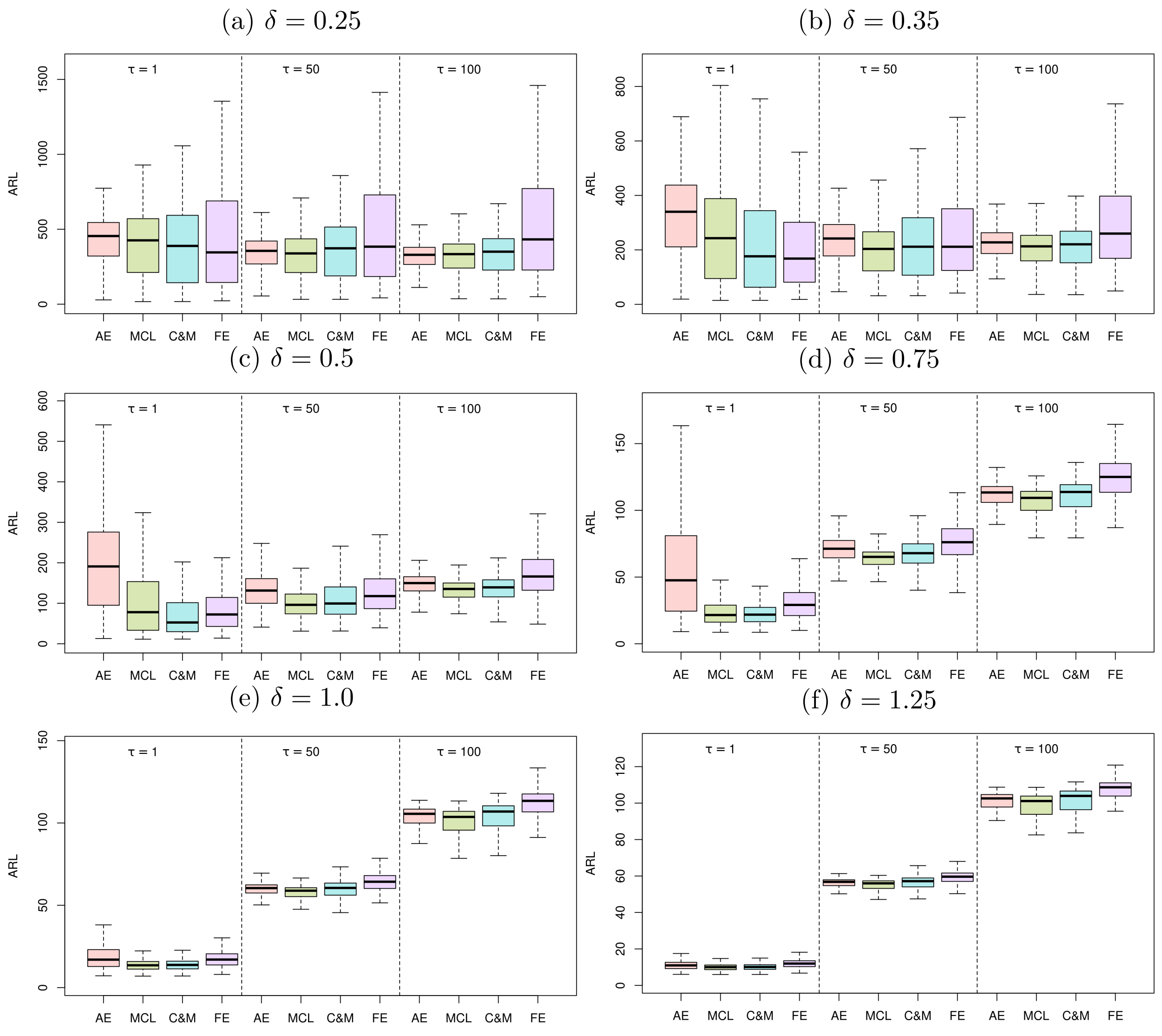


Figure 10: OC performance of the EWMA ($\lambda = 0.125$) control chart based on the fixed (FE), adaptive (AE) procedures, and two cautious learning parameter schemes, when $\theta = 4$. Control charts satisfy the GICP condition with $\beta = 0.1$. Boxplots are based on the 200 simulated conditional ARLs.

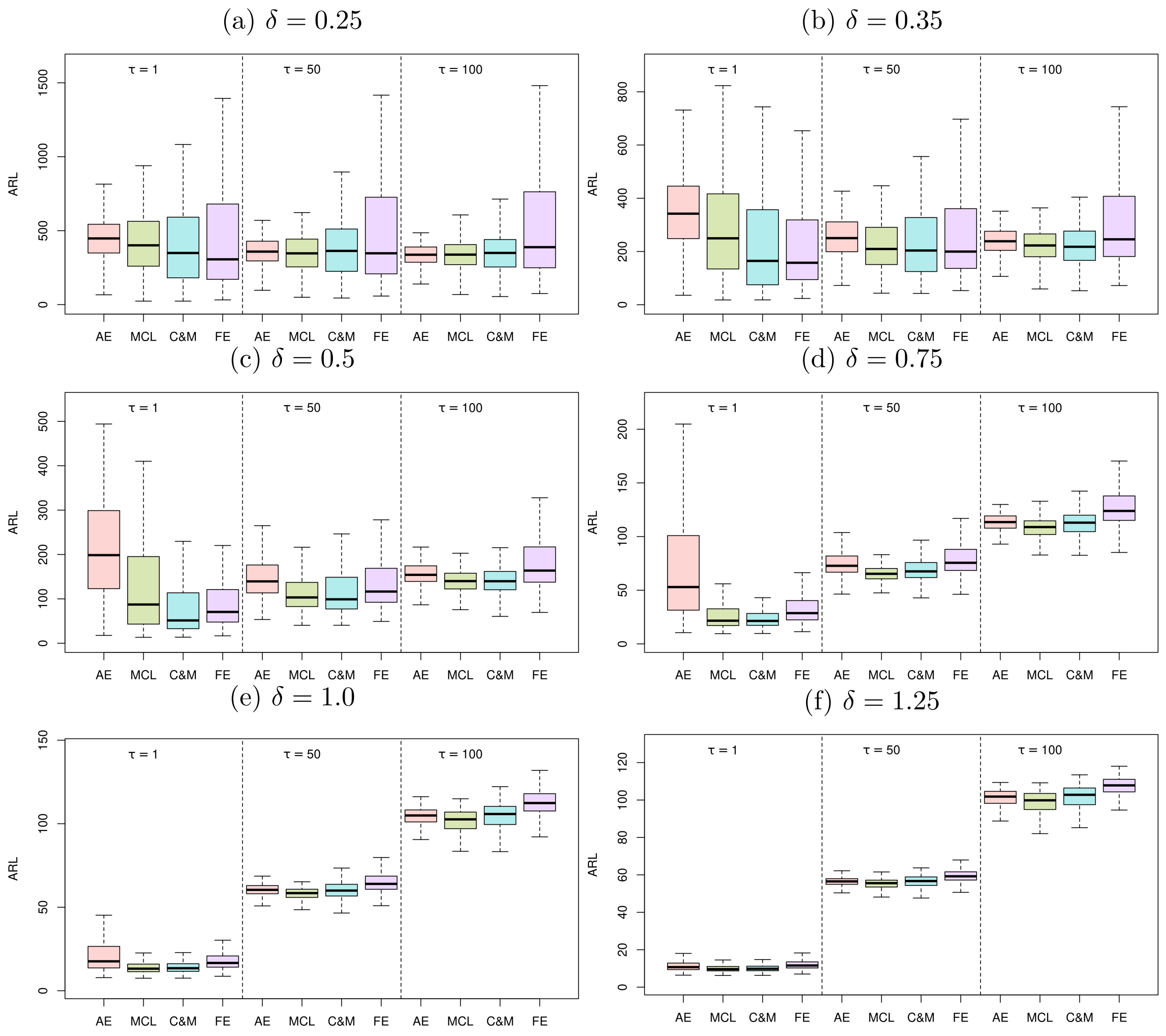


Figure 11: OC performance of the EWMA ($\lambda = 0.15$) control chart based on the fixed (FE), adaptive (AE) procedures, and two cautious learning parameter schemes, when $\theta = 4$. Control charts satisfy the GICP condition with $\beta = 0.1$. Boxplots are based on the 200 simulated conditional ARLs.

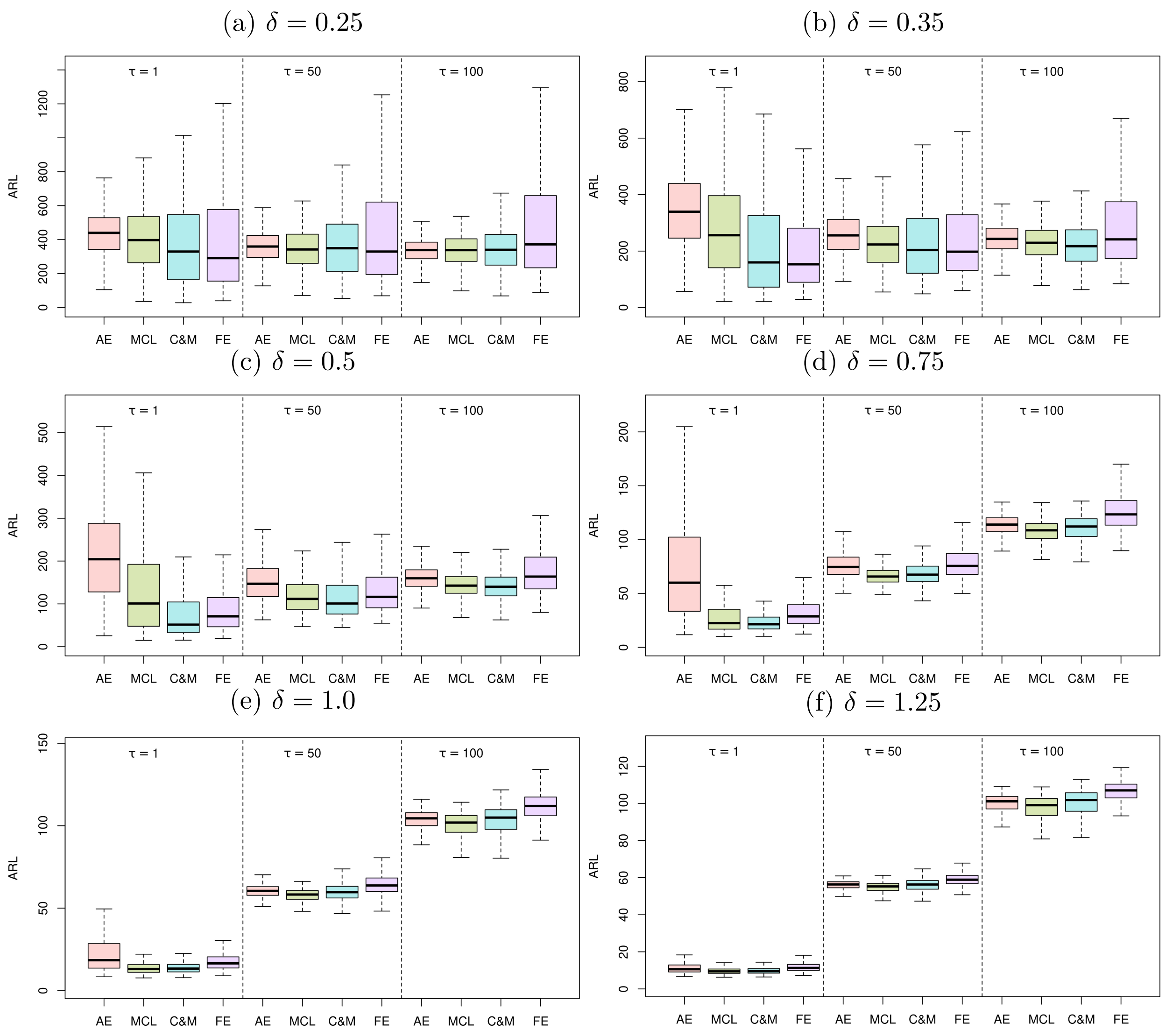


Figure 12: OC performance of the EWMA ($\lambda = 0.175$) control chartbased on the fixed (FE), adaptive (AE) procedures, and two cautious learning parameter schemes, when $\theta = 4$. Control charts satisfy the GICP condition with $\beta = 0.1$. Boxplots are based on the 200 simulated conditional ARLs.

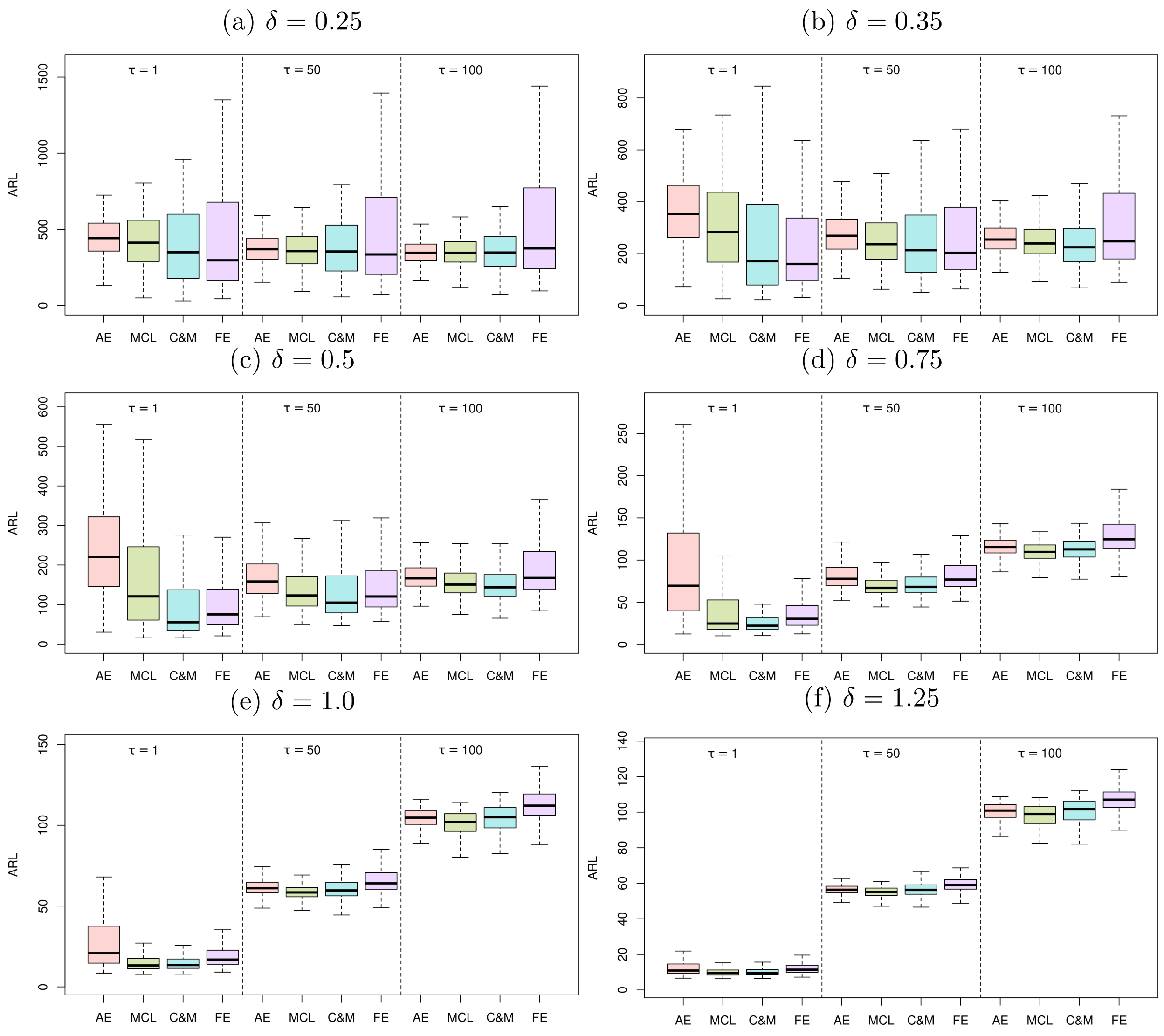


Figure 13: OC performance of the EWMA ($\lambda = 0.2$) control chart based on the fixed (FE), adaptive (AE) procedures, and two cautious learning parameter schemes, when $\theta = 4$. Control charts satisfy the GICP condition with $\beta = 0.1$. Boxplots are based on the 200 simulated conditional ARLs.

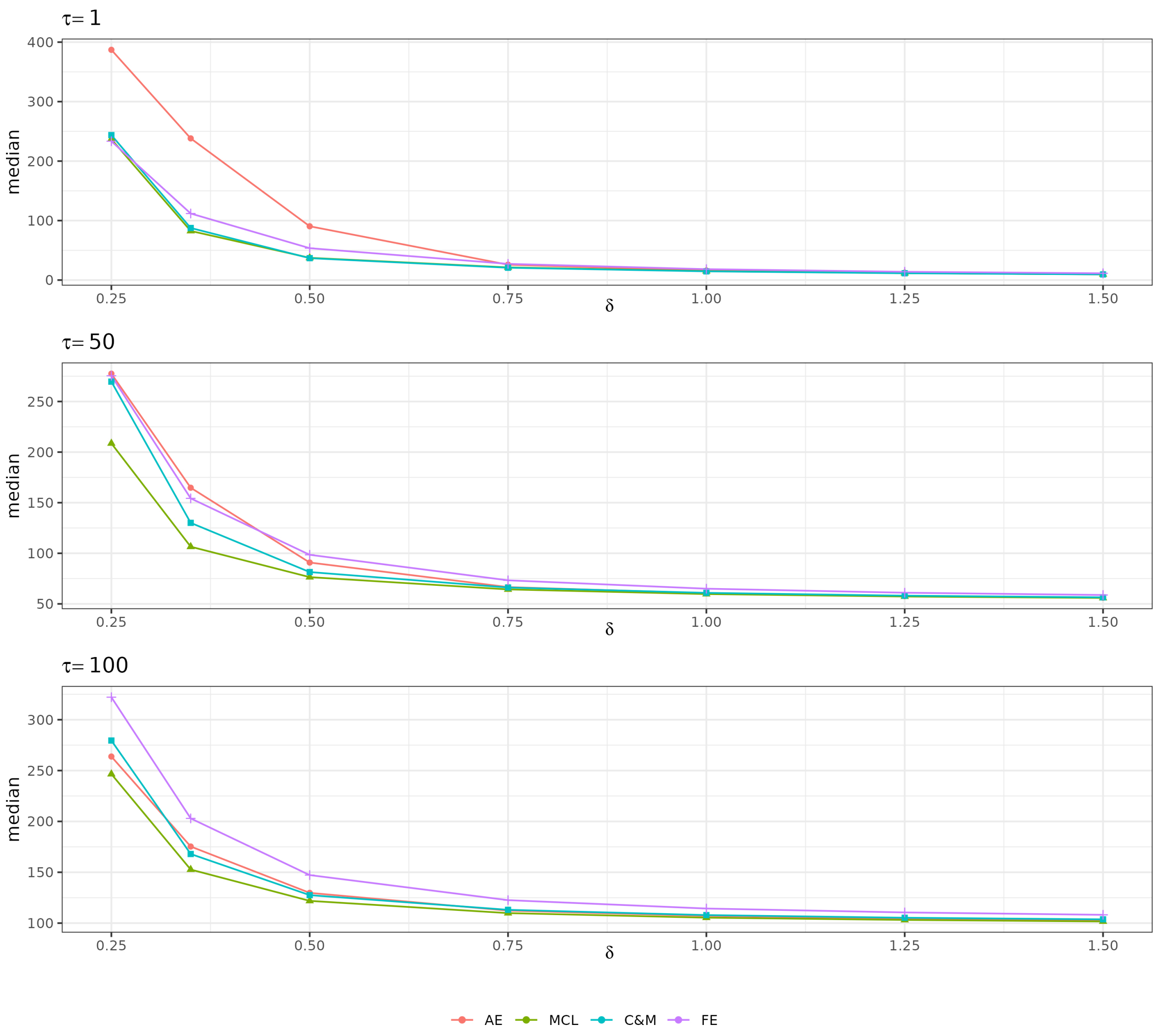


Figure 14: Median of the OC conditional ARL of the EWMA-type control chart under fixed (FE), adaptive (AE), cautious learning (CL) parameter updates for $\theta = 4$ and $\lambda = 0.05$. Control charts satisfy the GICP condition with $\beta = 0.1$. Plots are based on the 200 simulated conditional ARLs.

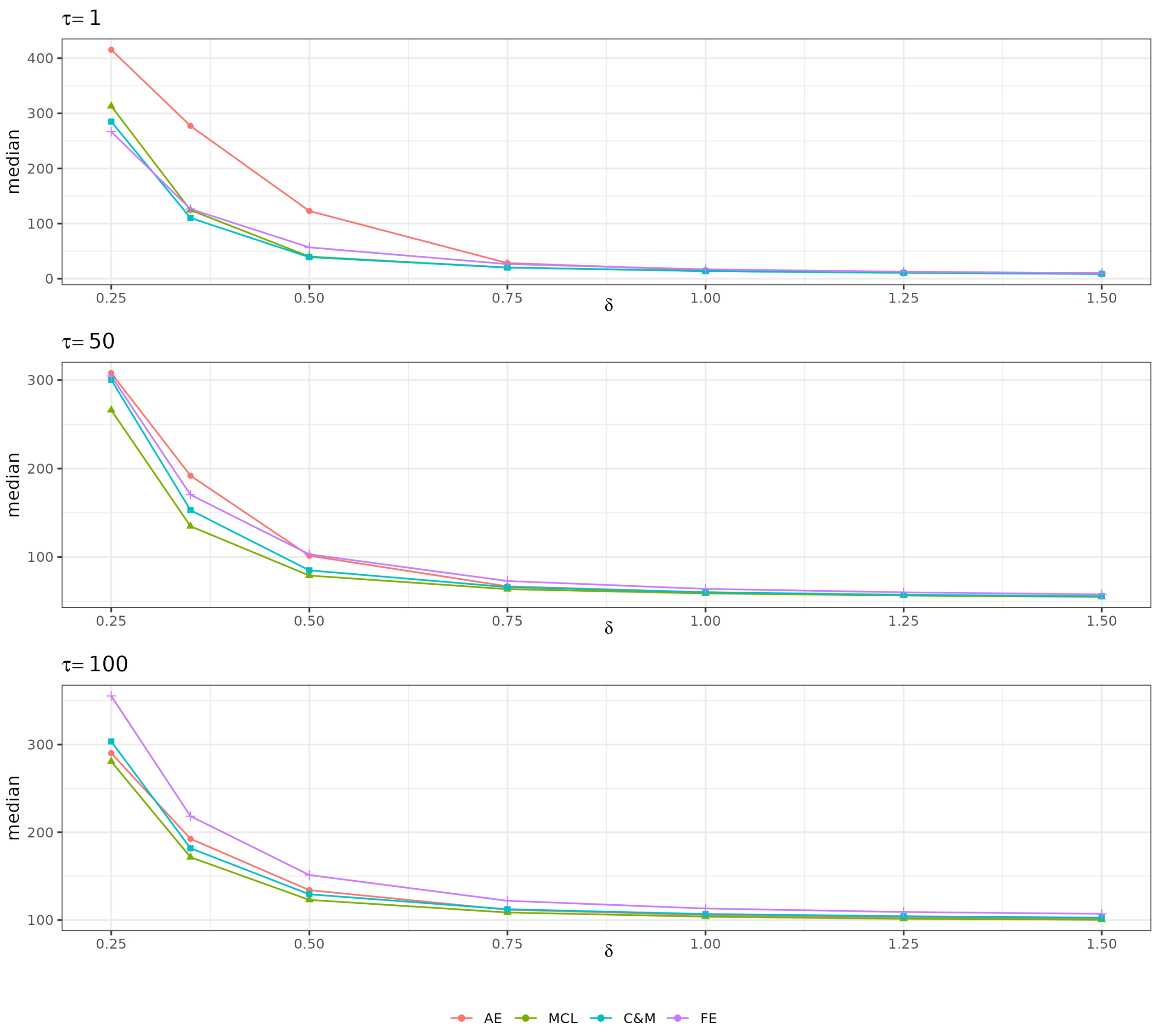


Figure 15: Median of the OC conditional ARL of the EWMA-type control chart under fixed (FE), adaptive (AE), cautious learning (CL) parameter updates for $\theta = 4$ and $\lambda = 0.075$. Control charts satisfy the GICP condition with $\beta = 0.1$. Plots are based on the 200 simulated conditional ARLs.

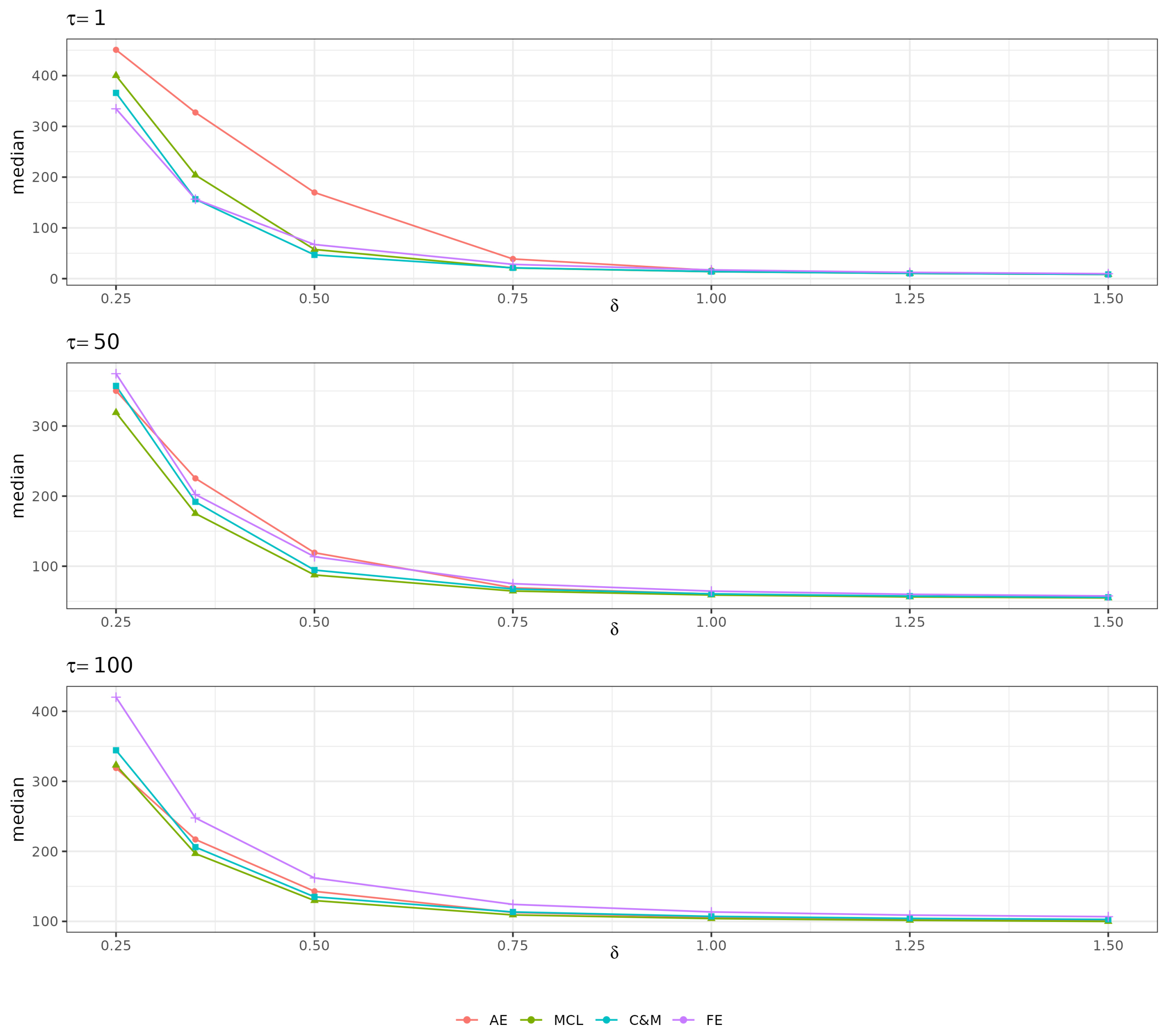


Figure 16: Median of the OC conditional ARL of the EWMA-type control chart under fixed (FE), adaptive (AE), cautious learning (CL) parameter updates for $\theta = 4$ and $\lambda = 0.1$. Control charts satisfy the GICP condition with $\beta = 0.1$. Plots are based on the 200 simulated conditional ARLs.

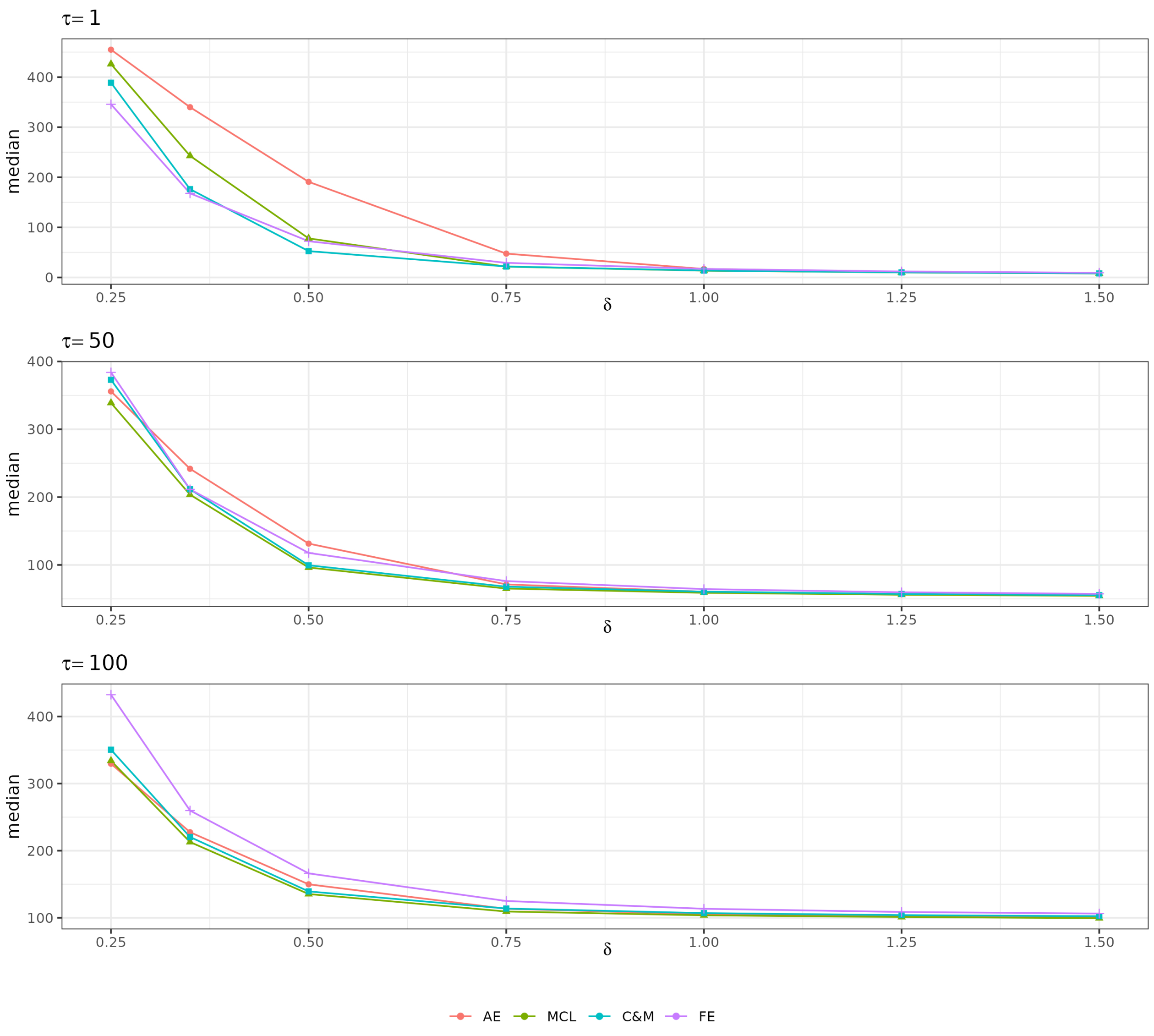


Figure 17: Median of the OC conditional ARL of the EWMA-type control chart under fixed (FE), adaptive (AE), cautious learning (CL) parameter updates for $\theta = 4$ and $\lambda = 0.125$. Control charts satisfy the GICP condition with $\beta = 0.1$. Plots are based on the 200 simulated conditional ARLs.

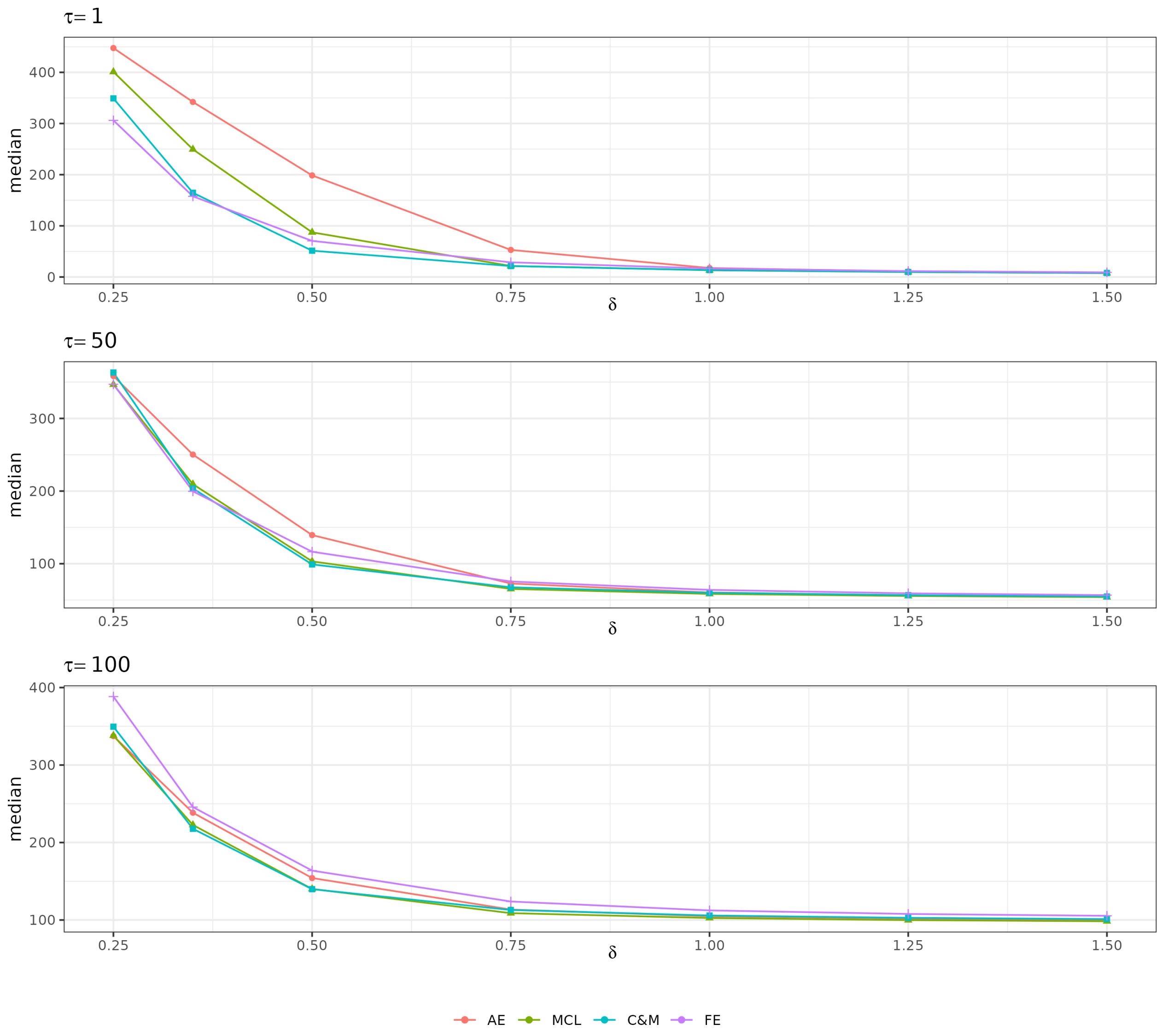


Figure 18: Median of the OC conditional ARL of the EWMA-type control chart under fixed (FE), adaptive (AE), cautious learning (CL) parameter updates for $\theta = 4$ and $\lambda = 0.15$. Control charts satisfy the GICP condition with $\beta = 0.1$. Plots are based on the 200 simulated conditional ARLs.

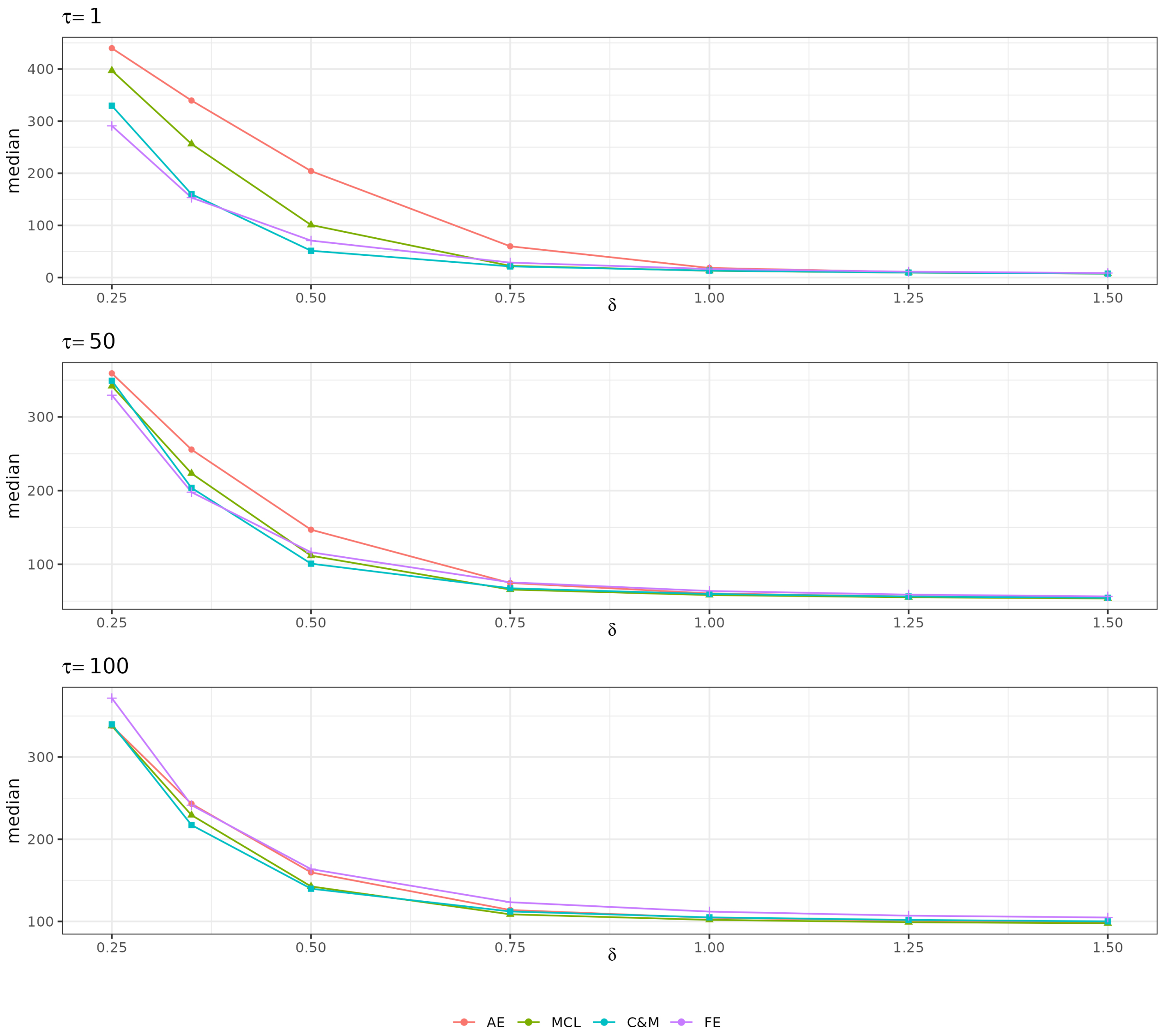


Figure 19: Median of the OC conditional ARL of the EWMA-type control chart under fixed (FE), adaptive (AE), cautious learning (CL) parameter updates for $\theta = 4$ and $\lambda = 0.175$. Control charts satisfy the GICP condition with $\beta = 0.1$. Plots are based on the 200 simulated conditional ARLs.

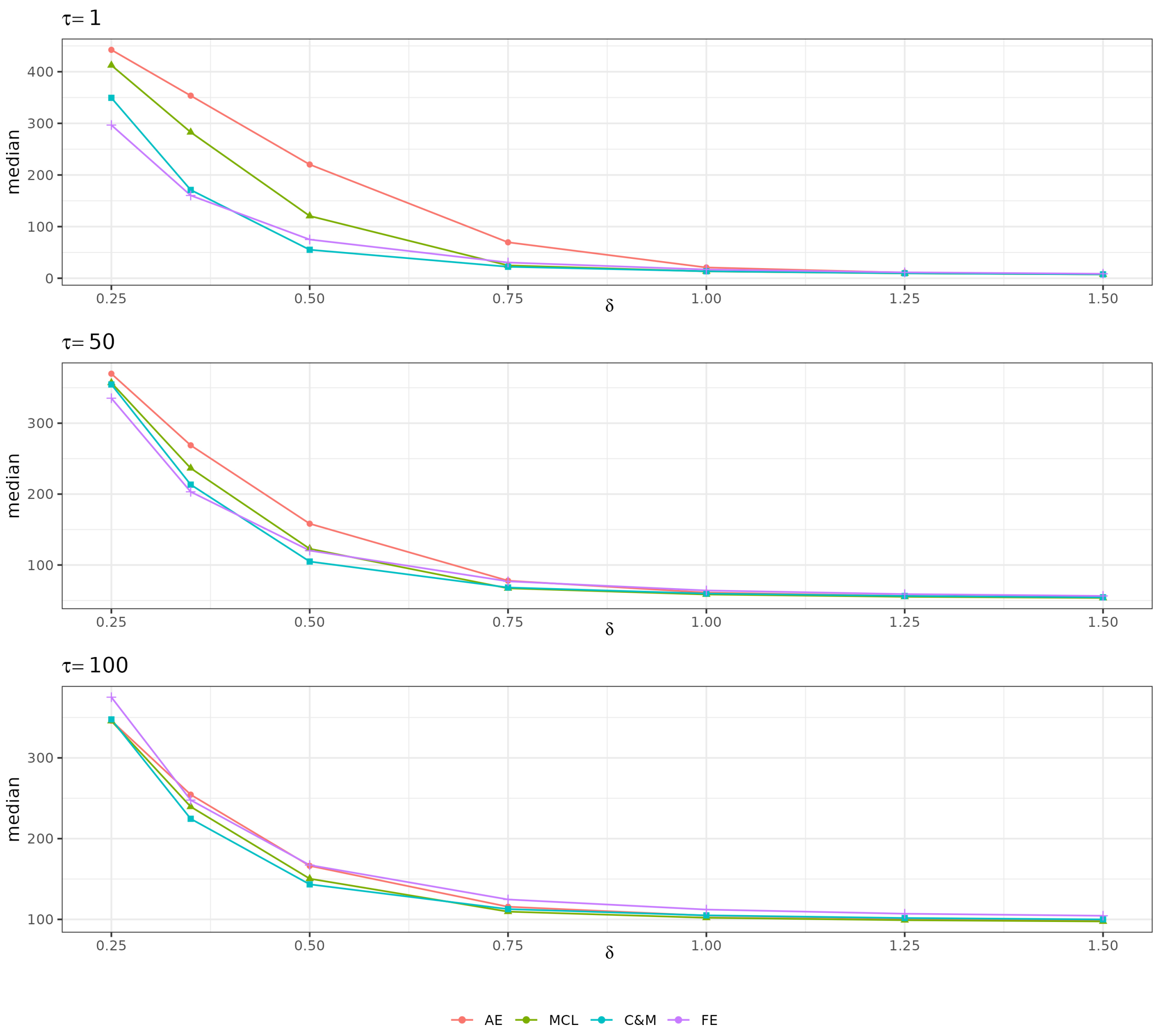


Figure 20: Median of the OC conditional ARL of the EWMA-type control chart under fixed (FE), adaptive (AE), cautious learning (CL) parameter updates for $\theta = 4$ and $\lambda = 0.2$. Control charts satisfy the GICP condition with $\beta = 0.1$. Plots are based on the 200 simulated conditional ARLs.

3 Application to the ICU admissions data

In the following, the code reproducing the analysis of the ICU dataset (in Section 5 of the main paper) is illustrated. The code is developed in the `Julia` programming language version 1.7, with some `R` code to produce the output tables. The simulations were carried out on a Linux Ubuntu 20.04 machine with a 11th Gen Intel Core i7-11800H 2.30GHz CPU and 32GB of RAM. The total computational time for this example is in the order of a minute (excluding code compilation).

First, load the necessary packages and the environment variables provided in the source code.

```julia
using DrWatson
#@quickactivate "CautiousLearning"
quickactivate("$(homedir())/Documents/git/SPC/CautiousLearning")

using Distributions, Random
using Parameters
using SharedArrays
using DataFrames, CSV, Dates
using Plots, StatsBase, StatsPlots, LaTeXStrings, RCall
using StatisticalProcessControl

include(srcdir("generate_data.jl"))
include(srcdir("update_parameter.jl"))
include(srcdir("simulate_runs.jl"))
include(srcdir("cfg.jl"))
```

Then, load the dataset located in the `data/ICUadmissions` folder. Here, data concerning the New York City area in 2020 are considered.

```julia
fold = "ICUadmissions"
dat = DataFrame(CSV.File(datadir(fold,
"New_York_Forward_COVID-19_Daily_Hospitalization_Summary_by_Region.csv")))
nydat = filter(row -> row.Region == "NEW YORK CITY", dat)
nydat.date .= Date.(nydat[:, 1], dateformat"mm/dd/yyyy")
nydat = filter(row -> year(row.date) == 2020, nydat);
```

Obtain the daily ICU counts along with the corresponding dates, starting from March 2020.

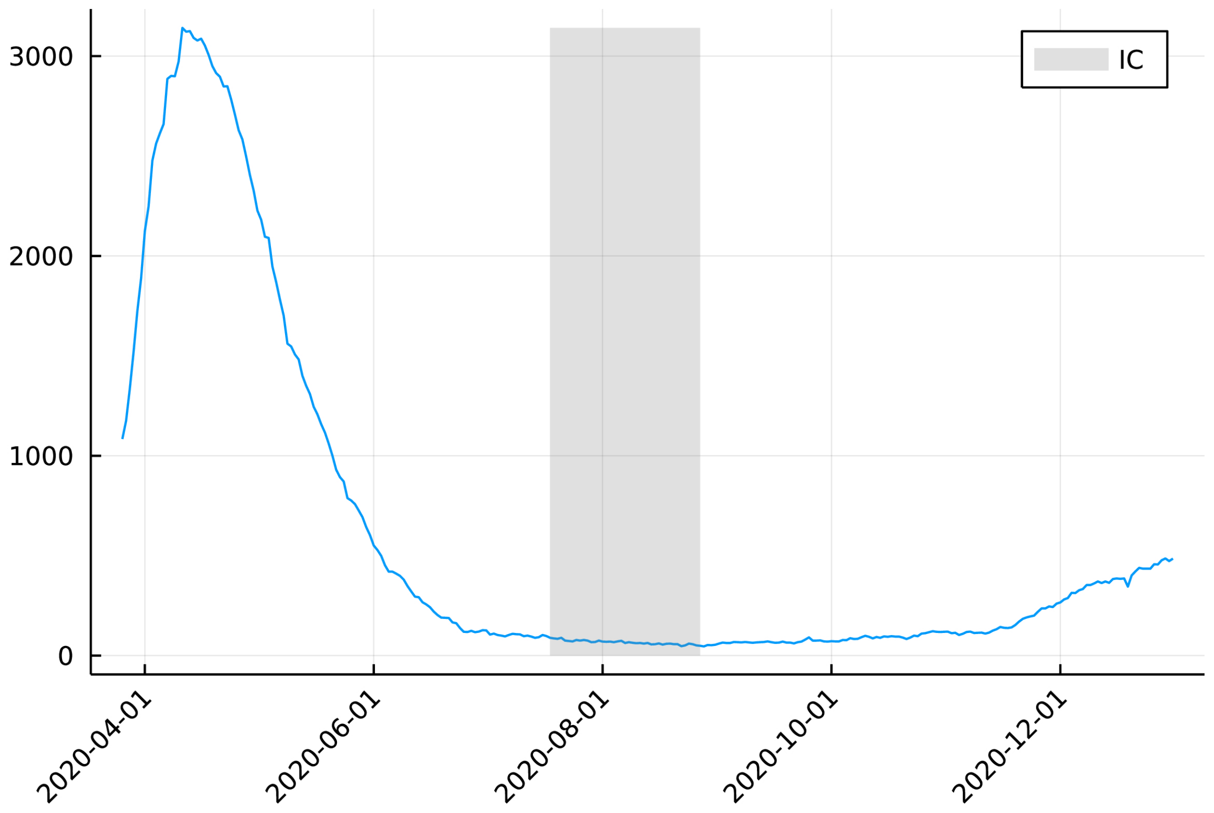
```julia
using Plots.PlotMeasures
y2020 = nydat[:, 4]
days2020 = nydat[:, 5]
first(y2020, 10)
first(days2020, 10)
```

```
10-element Vector{Dates.Date}:
 2020-03-26
 2020-03-27
 2020-03-28
 2020-03-29
 2020-03-30
 2020-03-31
 2020-04-01
 2020-04-02
 2020-04-03
 2020-04-04
```

Then, select the indices of the IC/OC datasets along with the initial sample size and the prospective monitoring size.

The following code reproduces Figure 6 of the main paper, which isolates the IC and OC data.

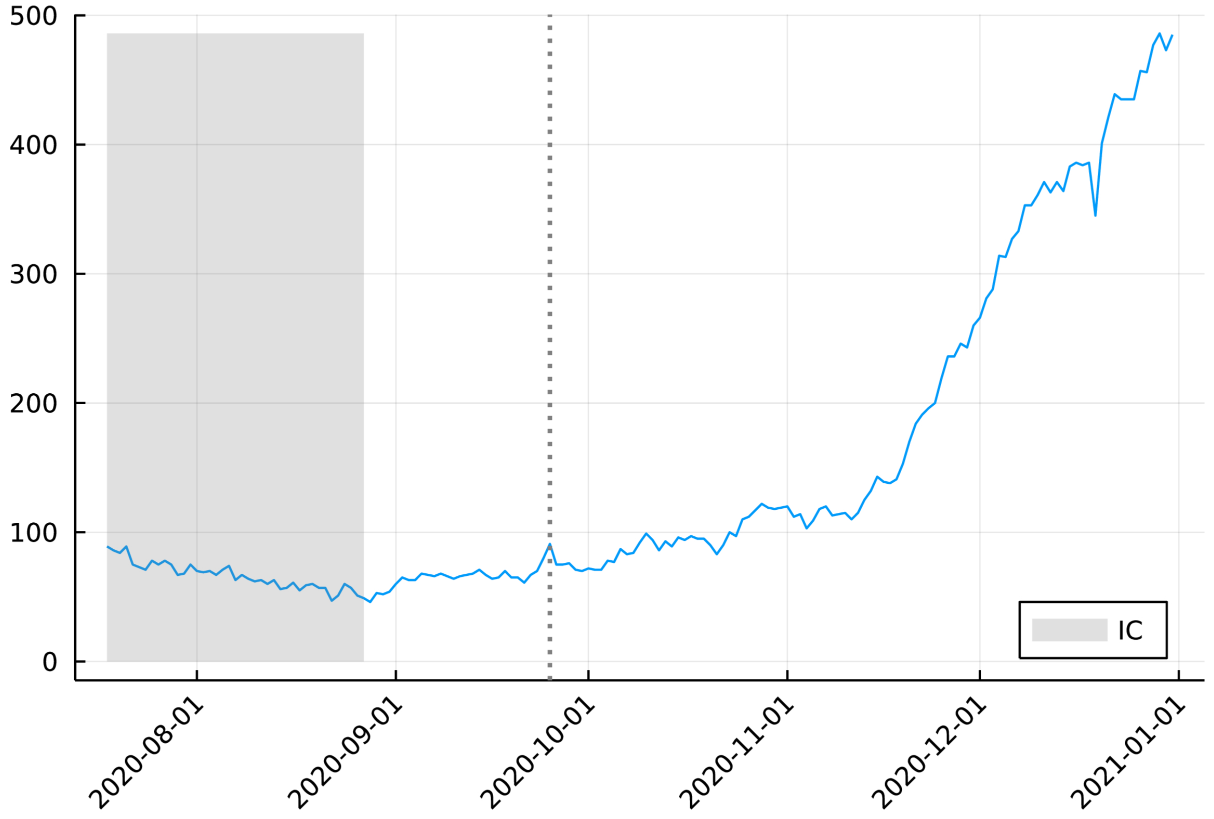
```
# ic_2020 = 115:175
ic_2020 = 115:155
oc_2020 = (ic_2020[end]+1):length(y2020)
n_ic = length(ic_2020)
n_oc = length(oc_2020)
pl = plot(days2020, y2020, label="", dpi=400, xrotation=45, bottom_margin=3mm)
plot!(pl, days2020[ic_2020], fill(0, n_ic), fillrange=fill(maximum(y2020), n_ic),
color=:gray, fillcolor = "gray", fillalpha=0.25, alpha=0.0, label="IC")
```



The following code reproduces Figure 7 of the main paper.

```
y = y2020[[ic_2020; oc_2020]]
days = days2020[[ic_2020; oc_2020]]

ic_idx = 1:n_ic
oc_idx = (n_ic+1):(n_ic+n_oc)
yIC = y[ic_idx]
daysIC = days[ic_idx]
yOC = y[oc_idx]
daysOC = days[oc_idx]

println("In-control data: ", daysIC[[1, end]])
pl = plot(days, y, label="", dpi=400, xrotation=45, bottom_margin=3mm)
plot!(pl, days[1:n_ic], fill(0, n_ic), fillrange=fill(maximum(y), n_ic), color=:gray,
fillcolor = "gray", fillalpha=0.25, alpha=0.0, label="IC", legend=:bottomright)
τ = n_ic + 29
vline!([days[τ]], color=:gray, linestyle=:dot, linewidth=2,  label="", markersize=2.5)
display(pl)

println("Possible change-point location: ", days[τ])
```

```
In-control data: [Dates.Date("2020-07-18"), Dates.Date("2020-08-27")]
Possible change-point location: 2020-09-25
```



Set the desired nominal in-control ARL to 500 and $\beta = 0.05$ for the GICP condition and calculate the desired nominal in-control ARL.

```
Arl0 = 500
beta = 0.05
thetaHat = mean(yIC)
println(thetaHat)
```

```
66.41463414634147
```

Define a function implementing the one-sided EWMA control chart using the various update mechanisms (AE, FE, MCL) defined in the source code.

```
function applyChart(ch, um, thetaHat, m, yprosp; seed = 123)
    # Random.seed!(seed)
    maxrl_i = length(yprosp)
    thetaHatVec = zeros(maxrl_i)
    thetaHatVec[1] = thetaHat
    di = 1
    diVec = Array{Int}(undef, maxrl_i)
    diVec[1] = di
    thetaHatCaut = thetaHat
    thetaHatCautVec = zeros(maxrl_i)
    thetaHatCautVec[1] = thetaHatCaut

    t_alarm = zeros(0)

    valueVec = zeros(maxrl_i)
    valueVec[1] = get_value(ch)
    i = 1
    while i < maxrl_i
        thetaHatCaut = thetaHatVec[i - di + 1]
        y = yprosp[i]
        ch = update_series(ch, chart_statistic(y, thetaHatCaut))
        thetaHat = update_parameter(thetaHat, y, i + m)
        i += 1
        valueVec[i] = get_value(ch)
        thetaHatVec[i] = thetaHat
        if check_update(ch, um)
            di = 1
        else
            di += 1
        end
        diVec[i] = di
        thetaHatCautVec[i] = thetaHatCaut
        if check_OC(ch)
            push!(t_alarm, i)
        end
    end

    return (t_alarm = t_alarm, dat = yprosp, chart_values = valueVec, limit_alarm =
get_limits(ch), limit_cautious = get_warning_limit(um), parameter_updates =
thetaHatCautVec, di = diVec)
end
```

```
applyChart (generic function with 1 method)
```

The function arguments are:

- `ch`: an `AbstractSeries` object, as defined in the `StatisticalProcessControl` package shipped with the source code.
- `um`: an `AbstractUpdate` object, as defined in the `StatisticalProcessControl` package shipped with the source code.
- `thetaHat`: estimate of the IC mean.
- `m`: initial sample size.
- `yprosp`: data onto which the control chart is applied.
- `seed`: eventual seed passed to the GICP optimization for reproducibility.

The function returns the following values:

- `t_alarm`: time of the first alarm.
- `dat`: data onto which the control chart has been applied.
- `chart_values`: vector of the EWMA control statistic values.
- `limit_alarm`: control chart limit.
- `limit_cautious`: the upper limit of the cautious region.
- `parameter_updates`: sequence of updated parameter estimates.

- di: sequence of d_t's.

Furthermore, a function applying the control chart is defined for monitoring data after computing the control limit which satisfies the GICP condition.

```
function applyChartGICP(ch, um, yinit, yprosp, thetaHat, Arl0; beta::Union{Bool,
Float64} = 0.2, maxrl=1e04, verbose=true, seed=Int(rand(1:1e06)))
    m = length(yinit)
    if isa(um, CautiousLearning)
        Ats0 = get_ATS(um)
        if Ats0 != 0
            # Calculate limit if Ats0 != 0, otherwise use zero-restarting chart
            if verbose println("Calculating limits for target ATS...") end
            sa_ats = saControlLimits(ch, AdaptiveEstimator(), runSimulation, Ats0,
thetaHat, Poisson(thetaHat),
                                    m, verbose=false, Amin=0.1, maxiter=1e05,
                                    gamma=0.015, adjusted=true, seed=seed)
            um = CautiousLearning(L = sa_ats[:h], ATS = Ats0)
        else
            if verbose println("ATS = 0, skipping limit calculation.") end
        end
        # Estimate cautious learning limit
    end

    if beta == false
        chart = deepcopy(ch)
    else
        chart = adjust_chart_gicp(ch, um, yinit, thetaHat, runSimulation, m, Arl0,
beta=beta, verbose=verbose)
    end

    return applyChart(chart, um, thetaHat, m, yprosp)
end
```

```
applyChartGICP (generic function with 1 method)
```

The GICP control limit correction uses the adjust_chart_gicp function, defined in the source code, implementing the computation of the control limit described in Subsection 4.1.1.

Finally, the EWMA control chart is applied using the FE, AE, and the MCL update rules. The following code reproduces Figure 8 in the main paper.

```
Random.seed!(2022-08-18)
D = Poisson
fname = plotsdir(fold, "alarms.jld2")

umVec = [CautiousLearning(ATS=0), FixedParameter(), AdaptiveEstimator()]
nms = ["MCL", "FE", "AE"]
perf = []
L = []
plts = []
for i in eachindex(umVec)
    ch = signedEWMA(l=0.2, L = 1.0)
    um = umVec[i]
    res = applyChartGICP(ch, um, yIC, yOC, thetaHat, Arl0, beta=beta)
    append!(L, res[:limit_alarm])
    plotsave = plotsdir(fold, nms[i]*".png")
    pl = plot(res.chart_values[1:55], label=L"C_t", legend=:outerright, dpi=400)
    hline!([res.limit_alarm], style=:dash, colour="red", xlab=L"t", ylab=L"C_t",
label="")
```

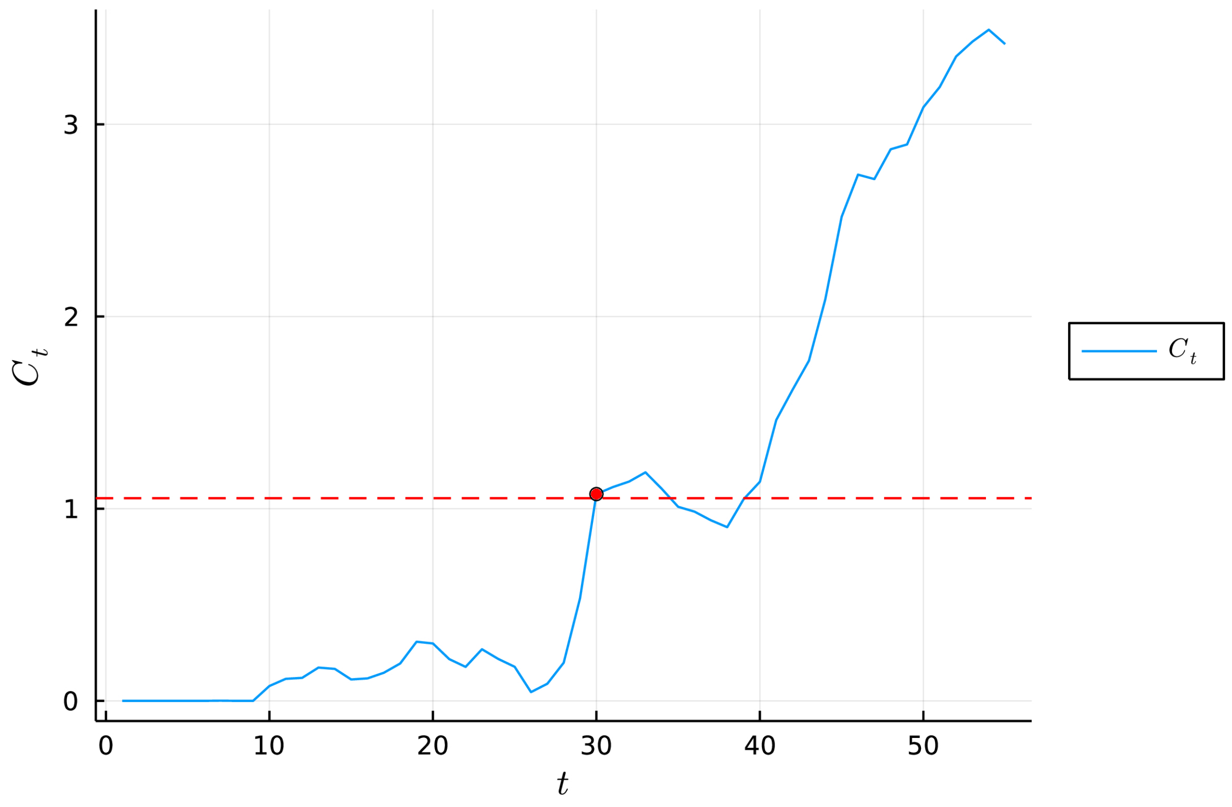
```
    tau = Int(first(res.t_alarm))
    scatter!([tau], [res.chart_values[tau]], colour="red", label="")
    append!(perf, tau)
    append!(plts, pl)
    println(tau,"\t", daysOC[tau])
end
display(plts[1])

@rput nms
@rput perf
@rput L

tab = R"""
library(knitr)
library(kableExtra)
df = cbind(nms, perf, L)
tex_OC <- kable(df, format="latex", digits = 2, row.names=FALSE,
col.names=c("Estimator", "Alarm", "Limit"), escape=FALSE, align='c', linesep = "",
    caption = "Time to alarm of the one-sided EWMA control chart using the fixed (FE),
adaptive (AE) and the proposed cautious learning (MCL) update rules.", label="ICU OC
alarm")
""" |> rcopy

println(tab)
```

```
ATS = 0, skipping limit calculation.
Calculating GICP for extremum 1/1...
GICP done.
30      2020-09-26
Calculating GICP for extremum 1/1...
GICP done.
41      2020-10-07
Calculating GICP for extremum 1/1...
GICP done.
31      2020-09-27
```

Furthermore, Table 1 below reproduces Table 3 of the main paper (before rounding the control limits).

Table 1: Time up to alarm of the one-sided EWMA control chart using the fixed (FE), adaptive (AE) and the proposed cautious learning (MCL) update rules.

Estimator	Alarm	Limit
MCL	30	1.05469060798468
FE	41	1.2147825075143
AE	31	1.02214273823991



References

Aly, A. A., N. A. Saleh, and M. A. Mahmoud (2021). An Adaptive Exponentially Weighted Moving Average Control Chart for Poisson Processes. *Quality Engineering 33*(4), 627–640.