r t Anno 2010_Numero 164

rapporti tecnici INGV

Tuning an Earthworm Phase Picker: Some Considerations on the *Pick_ew* Parameters

164

Istituto Nazionale di Geofisica e Vulcanologia

Anno 2010_Numero 164

rapporti tecnici

INGV

TUNING AN EARTHWORM PHASE PICKER: SOME CONSIDERATIONS ON THE *PICK_EW* PARAMETERS

Franco Mele, Andrea Bono, Valentino Lauciani, Alfonso Mandiello, Carlo Marcocci, Stefano Pintore, Matteo Quintiliani, Laura Scognamiglio and Salvatore Mazza

INGV (Istituto Nazionale di Geofisica e Vulcanologia, Sezione Centro Nazionale Terremoti)

164

Index

Introduction

Earthworm is one of the most widespread tools, in the seismological community, for real time processing of regional network data. Thanks are due to Carl Johnson who gave the first set of principia *Earthworm* development is based on, as scalability, flexibility, easy real-time data exchange, modularity and support [Johnson et al., 1995]. Although its first realization goes back to 1993, the fortune of *Earthworm* has been and is still in progress: since then, more than forty seismic networks all around the world have chosen *Earthworm* as a tool for real-time seismic monitoring. Earthworm users can rely today on the support of a vast community of seismologists daily involved in running it. In 1996 the Berkeley Seismographic Station (BSS) and the USGS, Menlo Park, formed a joint earthquake notification system for northern and central California, in which the *Earthworm* system provides fast locations and coda magnitudes, while the *REDI* (*Rapid Earthquake Data Integration*) system, triggered by *Earthworm*, computes local magnitudes and peak ground motions [Lind et al.,1996]. In the late '90s the *Earthworm* software was chosen to integrate data in the framework of the *United States National Tsunami Hazard Mitigation Program* [Oppenheimer et al., 2005]. In recent years a new impulse came to *Earthworm* from the collaboration between the USGS, CERI (Univ. of Memphis) and ISTI (Instrumental Software Technologies, Inc.) [ISTI, 2004]; nowadays ISTI provides consulting support for clients using *Earthworm* (and other earthquake data acquisition systems).

Although the *Earthworm* project started with the initial mission of replacing the old real time picker by Rex Allen, as developed by Jim Ellis [Allen, 1978; Allen, 1982; USGS et al., 2010], that picker is still in use in many networks. The name of the procedure that implements the Allen's algorithm is *pick_ew*. Despite its long-lived use, configuring this picker still seems not so immediate and easy. We discuss here a few simple rules to define some of the numerous (18) parameters of *pick_ew*, interpreting their physical meaning. Figure 1 reproduces a short example of the file that specifies, channel by channel, the *pick_ew* parameters. This paragraph reports some considerations we made during our effort in tuning those parameters. We hope that they can be useful to other researchers facing the same problem.

```
------------------------------------------------------


#
#  This is a sample station list for the pick_ew program.
#
#                                MinBigZC      RawDataFilt    LtaFilt        DeadSta         PreEvent
# Pick  Pin    Station/       MinSmallZC    MaxMint          StaFilt       RmavFilt         AltCoda
# Flag  Numb   Comp/Net/Loc   Itr1   MinPeakSize  i9  CharFuncFilt  EventThresh        CodaTerm        Erefs   ClipCount
# ----  ----   --------   ------------------------------------------------------------------------------------------
    1     0  AAR   VHZ NC  -- 3  40  3  60  500  3  .985  3.  .6  .03  5.  .9961  1200.  49.14  .8  1.5  50000.     2048
    1     1  AAS   VHZ NC  -- 3  40  3  60  500  3  .985  3.  .6  .03  5.  .9961  1200.  49.14  .8  1.5  50000.     2048
    0     2  ABL   VHZ CI  -- 3  40  3  60  500  3  .985  3.  .6  .03  5.  .9961  1200.  49.14  .8  1.5  50000.     2048
    1     3  ABR   VHZ NC  -- 3  40  3  60  500  3  .985  3.  .6  .03  5.  .9961  1200.  49.14  .8  1.5  50000.     2048
    1  2005  KCPB  BHZ NC  -- 3  40  3 162  500  3  .939  3.  .4  .015 5.  .9961  1200. 132.7   .8  1.5 135000.  8388608
# end of station file
------------------------------------------------------
```

Figure 1. An example of *pick_ew* parameters, as proposed by the original Earthworm documentation. The picture depicts the content of part of a *pick_ew.sta* file, containing the picker parameters on a channel by channel basis.

Our first sources of information are the *Earthworm* documentation [USGS et al., 2010], the very useful report: *Suggested Picker Parameter Changes,* by James C. Pechmann [1998] with its second revision [2006], the papers by Rex Allen [1978, 1982] and the analysis of the C program *pick_ew* itself. Our main intention was to understand the dimensions and physical meaning of some of the parameters. The suggested values are the result of our experiments on signals coming from the Italian National Seismic network of INGV; they depend on the configuration of the network, on the minimum magnitude of completeness that has been chosen as target of the network, and on the noise level of the territory monitored by the network. A different network configuration or target magnitude could lead to different best choices in the parameters.

1. Pick Flag, Pin Numb, SCNL (Station Component Network Location)

A parameter file, usually named *pick_ew.sta*, contains, for each channel, the parameters that regulate the behavior of the Allen algorithm on each channel (the name of the file containing the channel-picking

parameters can be changed in the control file specified in the starting command, i.e. in file *pick_ew.d*). All the parameters cannot be blank and are separated by blanks.

The first column of the *pick_ew.sta* file reports the *Pick Flag*. If *Pick Flag* is 0, *pick_ew* will not try to pick P-wave arrivals from this trace. If *Pick Flag* is 1, the trace will be picked. At column two we find the Pin Number; it is not used by *pick_ew*. Nevertheless it has to be unique all over the entire system. At columns 3, 4, 5 and 6 we find the so called "SCNL" group of parameters. SCNL is a concatenation of Station code (3 to 5 characters), Component (3 characters), Network (2 characters) and Location (2 characters) that uniquely identifies a channel. For details on SCNL and its rules, please see the official documentation (http://folkworm.ceri.memphis.edu/ew-doc/cmd/pick_ew_cmd.html).

2. The Rex Allen algorithm in pick_ew: how to declare the onset of a phase

The basic rules for many triggering algorithms are as follows:

1. define a (positive) characteristic function of the signal that can be computed at each new sample;
2. recursively compute, at each new sample, a Long Term Average (LTA) of the characteristic function; this LTA is a synthetic description of the background noise in one unique number (or a reduced set of numbers);
3. recursively compute, at each new sample, a Short Term Average (STA) of the characteristic function; this STA is supposed to be very similar to the LTA when the signal is only due to the background noise; but STA is supposed to very quickly become very different from LTA when the seismic signal begins.
4. The ratio STA/LTA is close to 1 when the signal is in the background noise. A "pick" is declared when the STA/LTA ratio goes over a predefined threshold.

We look at the *sample.c* (part of *pick_ew*) program and at the Allen first paper together. The characteristic function *E(t)* proposed by Allen [1978] and used in *sample.c* is:

$$E(t) = f(t)^2 + C_2 f'(t)^2 \; ; \qquad f'(t) = f(t) - f(t-\Delta t)$$

where $f(t)$ is the input signal after high-pass filtering (to remove DC offset), $f'(t)$ is the first difference of the high-pass filtered signal $f(t)$, and Δt is the sampling interval.

The pseudo-code from Allen is:

1. High-pass filter the input signal, to remove DC offset, with characteristic parameter C_1, obtaining the filtered time series $f(t)$.
2. Calculate the first difference $f'(t) = f(t) - f(t-\Delta t)$.
3. Calculate the characteristic function $E(t) = f(t)^2 + C_2 f'(t)^2$.
4. Calculate the STA of $E(t)$, using a low-pass filter with parameter C_3 .
5. Calculate the LTA of $E(t)$, using a low-pass filter with parameter C_4 .
6. Calculate the threshold level $Y(t) = C_5 \, \mathrm{LTA}\{E(t)\}$
7. If $\mathrm{STA}\{E(t)\} > Y(t)$, declare an event.

The correspondences between the Allen symbols and those used in *sample.c* are as follows:

f(t): high-pass filtered sample of the signal	*rdat*
E(t): characteristic function time series	*edat*
Short term average time series	*esta*
Long term average time series	*elta*
Threshold level	*eref*
C_1	*RawDataFilt*
C_2	*CharFuncFilt*
C_3	*StaFilt*
C_4	*LtaFilt*
C_5	*EventThresh*

3. Waveform Filtering Parameters

3.1 CharFuncFilt

As Allen states, "*C_2 [CharFuncFilt] is a weighting constant whose purpose is to vary the relative weight assigned to amplitude and first difference, depending on digital sample rate and noise characteristics of the individual seismic station*". Actually *CharFuncFilt* is the multiplier of the squared first difference $f'(t)^2$ of the signal, summed, in the definition of the characteristic function, with $f(t)^2$, the squared signal itself. $f'(t)$ is proportional to the (approximate) derivative of the signal (to the acceleration if the network uses velocimeters only) but is not the actual acceleration: it has the same dimension of the signal. We are using today the value 3 for all the stations of the Italian National Seismic Network (as found in the example of *Earthworm* and in the documentation), but what follows would suggest a more careful choice.

Being $E(t)$ the characteristic function:

$$E(t) = f(t)^2 + C_2 f'(t)^2 = f(t)^2 + C_2 [f(t) - f(t-\Delta t)]^2$$

in order to define a characteristic function in a way that is independent of the sampling rate, we should state:

$$f^{*\prime}(t) \approx [f(t) - f(t-\Delta t)]/\Delta t$$

$$E(t) = f(t)^2 + C^*_2 [f(t) - f(t-\Delta t)]^2/\Delta t^2 = f(t)^2 + C^*_2 f^{*\prime}(t)^2 \quad \text{where } C^*_2 = C_2 \Delta t^2$$

In this way $E(t)$ is a function of the signal $f(t)$ and of its derivative $f^{*\prime}$ and is not a function of the sampling rate. Therefore, for a consistent choice, we should choose, for each channel with different sampling rates Δt:

$$C_2 = C^*_2 / \Delta t^2.$$

As an example, having,

$$C_2 = 3 = C^*_2/(0.0001\ s^2)$$

for a channel sampled at 100 sps, as suggested by the *Earthworm* documentation, we would have

$$C^*_2 = 0.0003\ s^2$$

A consistent choice for all the channels of the network of any sampling rate $sps = 1/\Delta t$ would be:

$$CharFuncFilt = C_2 = C^*_2 / \Delta t^2 = (0.0003\ s^2) / \Delta t^2.$$

3.2 StaFilt and LtaFilt

StaFilt and *LtaFilt* are the main parameters that tune the computation of the STA/LTA (Short-Term-Average/ Long-Term-Average). To understand their physical meaning, a short digression on electric low-pass RC filters is convenient. Figure 2 shows an RC circuit, where U is the input signal and V_t is the output. If the capacitor C is initially uncharged ($V_{t-\Delta t} = 0$), the charge of the RC circuit follows the relation:

$$V_t = U (1 - e^{-t / RC}).$$

The cell $V_{t-\Delta t}$ symbolizes the initial charge of the capacitor; when it is not null, the charge of the RC circuit at time t is:

$$V_t = (U - V_{t-\Delta t}) (1 - e^{-\Delta t / RC}) + V_{t-\Delta t}$$

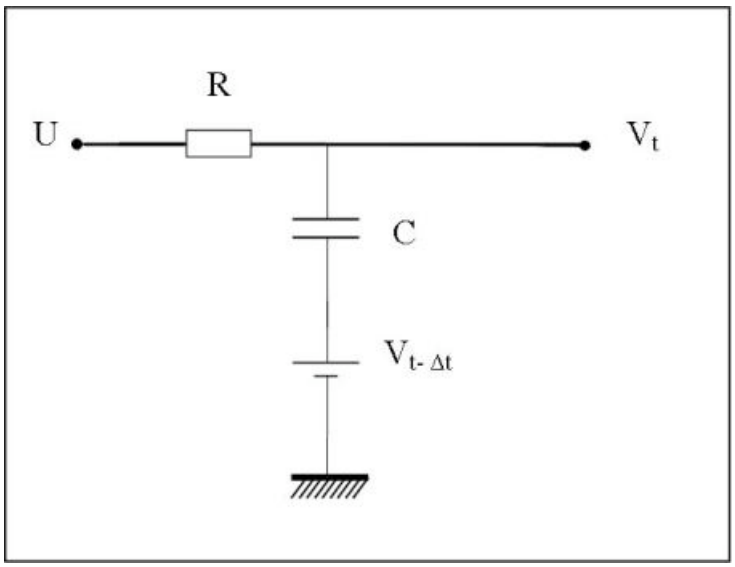

Figure 2. Sketch of an RC circuit. The initial charge of the capacitor C is represented by the cell $V_{t-\Delta t}$. U is the input signal while V_t is the filtered output at time t.

If we state:

$$\alpha = (1 - e^{-\Delta t / RC}) \qquad \text{or}$$
$$\alpha = (1 - e^{-\Delta t / \tau})$$

we can write:

$$V_t = V_{t-\Delta t} + \alpha (U - V_{t-\Delta t}) \qquad (1)$$

It is also useful to remind the following relations for the cut-off frequency f_c of a low-pass RC circuit:

$$f_c = 1/(2 \pi RC)$$

$$1 /RC = 1/ \tau = 2 \pi f_c = 2 \pi / T$$

Hence we have:

$$\alpha = (1 - e^{-2 \pi f_c \Delta t}) = (1 - e^{-2 \pi \Delta t / T}).$$

Going back to *pick_ew*, in the file *sample.c* it is possible to find two apparently different ways of low-pass filtering; the first is used twice to compute the short-term average *Sta->esta* and the long-term average *Sta->elta*; the second formula is used to compute *eabs*, the running mean absolute value of *rdat*.
From *sample.c* we read:

```
/* … omissis … */
/* Compute esta, the short-term average of edat */
  Sta->esta += Parm->StaFilt * (edat - Sta->esta);

/* Compute elta, the long-term average of edat */
  Sta->elta += Parm->LtaFilt * (edat - Sta->elta);
```

```
/* ... compute eref (omissis) … */

/* Compute eabs, the running mean absolute value of rdat */
  Sta->eabs = (Parm->RmavFilt * Sta->eabs) +
            (( 1.0 - Parm->RmavFilt ) * fabs( Sta->rdat ));
}
/* end of sample.c */
```

edat is the characteristic function, that is, the input signal of the first two filters, while the absolute value of *rdat* is the input function of the third filtering operation, used to compute the duration of the signal (*rdat* is the true signal after high-pass filtering). We can rewrite the first recursive filtering algorithm as:

$$L(i) = L(i-1) + \alpha \; [\, X(i) - L(i-1) \,] \qquad (2)$$

where X(i) is the input value and L(i) is the low-pass filtered result. L(i-1) is the filtered result at the previous time-step. This equation is exactly the relation (1) of the RC low-pass filter. Let us call $T = 2\pi\,\tau = 2\pi\,RC$ the characteristic "time of charge" of the circuit, or the characteristic time of integration of the low-pass filter; the parameter α of equation (1) can be expressed as:

$$\alpha = (1 - e^{-\Delta t / \tau_1}) = (1 - e^{-2\pi\,\Delta t / T_1}) = (1 - e^{-2\pi\,\Delta t\; fc_1}) \qquad (3)$$

The factor α stands for *StaFilt* (or *Parm->StaFilt*) in the first filter (used to produce the Short Term Average of the characteristic function); an analogous equation

$$\beta = (1 - e^{-\Delta t / \tau_2}) = (1 - e^{-2\pi\,\Delta t / T_2}) = (1 - e^{-2\pi\,\Delta t\; fc_2})$$

can be written for *LtaFilt* (or *Parm->LtaFilt*) in the second filter (used to produce the Long Term Average).

The physical meaning of α (or β) is suggested by equation (3) where Δt = 1/ sps is the sampling interval in seconds and $T = 2\pi\,\tau$ is the characteristic time of the filter expressed in seconds. An intuitive experiment to understand the meaning of T is to use an Heaviside step function $H(t)$ as input X(i) for the filter of equation (2) (see figures 3a and 3b):

$H(t) = 0 \; ; \; t < 0$
$H(t) = 1 \; ; \; t >= 0$

For $t = \tau$ the filtered output signal L(t) is about the 63.3% of the step function and for t = T the output signal is 99.8% of the input signal.

In this view we can rewrite:

$$\alpha = StaFilt = (1 - e^{-\Delta t / \tau_1}) = (1 - e^{-2\pi\,\Delta t / T_1}) \qquad (4)$$
$$\beta = LtaFilt = (1 - e^{-\Delta t / \tau_2}) = (1 - e^{-2\pi\,\Delta t / T_2}) \qquad (5)$$

Now the problem of defining the *StaFilt* and *LtaFilt* parameters is conceived as the definition of two characteristic time lengths T_1 and T_2.

3.2.1 Suggested StaFilt and LtaFilt

Taking the logarithm of equations (4) and (5) we can write:

$$T_1 = 2\pi\,\tau_1 = 2\pi\,\Delta t \,/\, \ln [1 \,/ (1 - StaFilt)]$$
$$T_2 = 2\pi\,\tau_2 = 2\pi\,\Delta t \,/\, \ln [1 \,/ (1 - LtaFilt)]$$

Following the previous suggestions in the Earthworm documentation for Short-Period instruments (1 second response) recorded at 100 samples per second, we set:

$T_1 = 0.068$ s (6)
$T_2 = 2.06$ s

which result in the two adimensional parameters at a sampling rate of 100 sps:

StaFilt = 0.6
LtaFilt = 0.03

StaFilt and *LtaFilt* have to be computed from equations (4), (5) and (6) to correct for other sampling rates.

Following the suggestions of Jim Pechmann, we chose for BB and VBB instruments in eq. (4) and (5):

$T_1 = 0.123$ s (7)
$T_2 = 4.19$ s

With this choices and $\Delta t = 0.01$, we obtain the parameters:

StaFilt = 0.4 and
LtaFilt = 0.015.

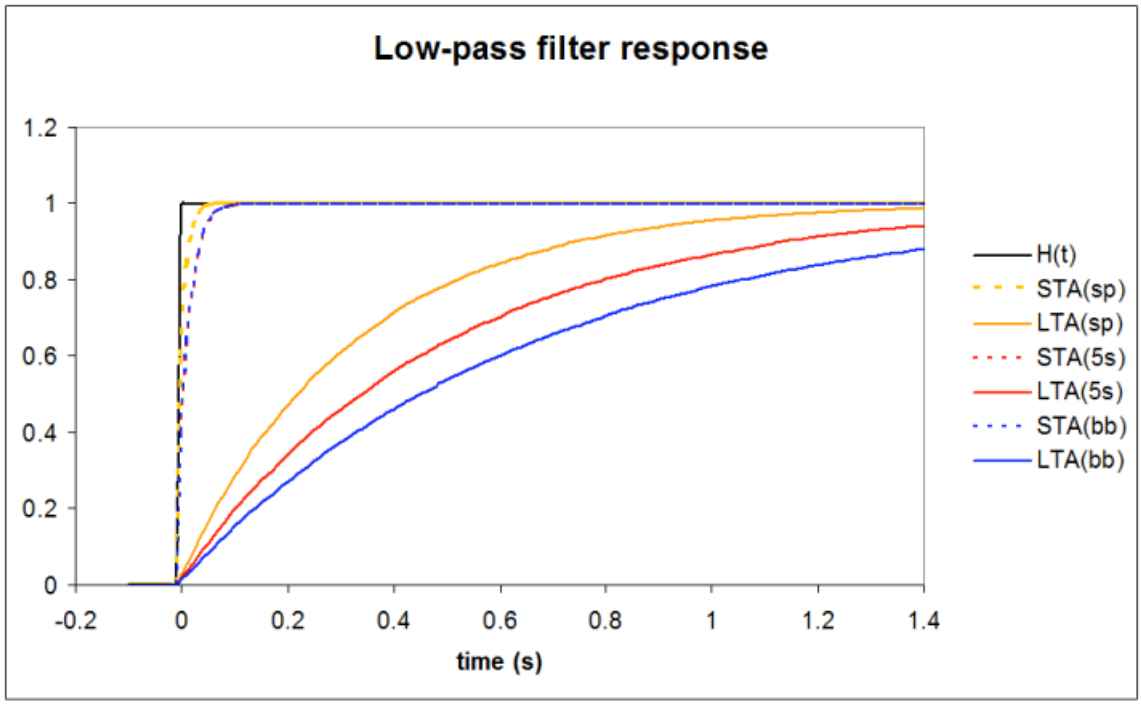

Figure 3. Responses of the low-pass filters (proposed in this report) to an Heaviside input function.

For BB and VBB data streams with different sampling rates, *StaFilt* and *LtaFilt* have to be computed from equations (4), (5) and (7).

For instruments like LE3D-5S, with flat response up to 5 seconds, we are testing these parameters:

$T_1 = 0.126$ s (8)
$T_2 = 3.14$ s.

With this choices and $\Delta t = 0.01$, we obtain the parameters:

StaFilt = 0.393 and
LtaFilt = 0.02.

3.3 RmavFilt

Going back to *sample.c* we read:

```
/* Compute eabs, the running mean absolute value of rdat */
  Sta->eabs = (Parm->RmavFilt * Sta->eabs) +
            (( 1.0 - Parm->RmavFilt ) * fabs( Sta->rdat ));
}
/* end of sample.c */
```

RmavFilt regulates the computation of *eabs*, the running mean absolute value of *rdat* through a low-pass filtering (for the definition of *rdat* see the following paragraph on *RawDataFilt*). The resulting *eabs* is used to define the duration of the seismic signal. The filtering algorithm can be rewritten:

$$L(i) = \zeta \; L(i-1) + (1- \zeta) \; X(i) \qquad (9)$$

where X(i) is the input function and L(i) and L(i-1) are the actual and previous value of the filtered function as in equation (2). Setting:

$$\zeta = 1 - \alpha \qquad (10)$$

equation (10) becomes:

$$L(i) = (1- \alpha) \; L(i-1) + \alpha \; X(i) = L(i-1) + \alpha \; [X(i) - L(i-1)]$$

Hence equations (2) and (9) are identical. Taking into account equations (3) and (10), the physical interpretation of *RmavFilt* is:

$$RmavFilt = e^{-\Delta t / \tau_3} = e^{-2 \pi \Delta t / T_3} \qquad (11)$$

where Δt is the sampling interval and T_3 is the characteristic time of the filter expressed in seconds. The Earthworm documentation suggests

$$RmavFilt = 0.9961$$

this value, for a signal sampled at 100 sps, corresponds to a characteristic time

$$T_3 = 16.08 \text{ s.}$$

For a different sampling interval Δt, *RmavFilt* has to be computed opportunely through equation (11).

3.4 RawDataFilt

Having written a low-pass filter, a straightforward way of obtaining as output an high-pass filtered signal H(i) would be to subtract the low-pass filtered signal L(i) from the input unfiltered signal X(i).

$$H(i) = X(i) - L(i) \qquad (12)$$
$$H(i-1) = X(i-1) - L(i-1).$$

L(i) is the output of a low-pass filter with X(i) as input. Using equations (2) and (12) we can write:

$$H(i) = X(i) - L(i-1) - \alpha \; [X(i) - L(i-1)]$$

and using (12):

$$H(i) = X(i) - X(i-1) + H(i-1) - \alpha \; [X(i) - X(i-1) + H(i-1)]$$
$$H(i) = (1 - \alpha) \; [X(i) - X(i-1) + H(i-1)].$$

Using equation (10) we obtain for a high-pass filter:

$$H(i) = \zeta \; [\, H(i-1) + X(i) - X(i-1) \,] \qquad (13)$$

The physical interpretation of ζ is (see (3) and (10)):

$$\zeta = e^{-\Delta t / \tau} = e^{-\omega \Delta t} = e^{-2\pi f_c \Delta t} \qquad (14)$$

Hence the corner frequency f_c of the high-pass filter is computed from ζ as:

$$\ln(\zeta) = -2\pi f_c \Delta t$$
$$f_c = -\ln(\zeta) / (2\pi \Delta t).$$

Going back to *sample.c*, we can verify that, in the *Earthworm* procedures, the high-pass filtering is obtained in a slightly different way:

```
/* … omissis … */
/* Store present value of filtered data */
  Sta->rold = Sta->rdat;
/* Compute new value of filtered data */
  Sta->rdat = (Sta->rdat * Parm->RawDataFilt) +
    (double) (LongSample - Sta->old_sample) + small_double;
/* … omissis … */
/* Store integer data value */
  Sta->old_sample = LongSample;
/* … omissis … */
```

LongSample is the current sample of the raw signal. More simply it can be written:

$$H_{EW}(i) = \chi \; H_{EW}(i-1) \; + [X(i) - X(i-1)]$$

which is a close approximation of equation (13). Nevertheless the parameter χ can be interpreted as the parameter ζ of equation (14), so that we can state:

$$RawDataFilt = e^{-2\pi fc \Delta t} \qquad (15)$$

where $\Delta t = 1/\text{sps}$ is the sampling time interval and f_c is the corner frequency of the high-pass filter we want to apply to the raw signal. Pechmann [1998 – 2006] suggests to filter the broad-band and very-broad-band signals at 1 Hz. For $\Delta t = 0.01$ s and $f_c = 1$ Hz we obtain from equation (15):

$$RawDataFilt = 0.939$$

as proposed by Pechmann [2006]. For short-period instruments, choosing $f_c = 0.24$ Hz and $\Delta t = 0.01$ s, equation (15) gives

$$RawDataFilt = 0.985$$

After some experiments we decided to adopt for the entire *Italian National Seismic Network*, and for any kind of velocimeter, a cutoff frequency f_c for the high-pass filter at 4.0 Hz that gave us the best results. Hence

$$RawDataFilt = 0.777$$

for all the channels sampled at 100 sps. For other sampling rates, we apply (15) with $f_c = 4$ Hz.

3.5 EventThresh

A trigger is declared in *pick_ew* if the short term average (of the characteristic function) *esta* goes over *EventThresh* times the long term average *elta*. The documentation suggests *EventThresh* = 5. Unfortunately (in a seismological point of view) Italy is surrounded by seas, with more than 7400 kilometers of coasts for a surface of only 300,000 squared kilometers. Many seismogenic sources are located in sea areas, but are nevertheless relevant for seismic risk; in order to ensure the location of low magnitude earthquakes we frequently have to rely on distant stations and weak signals. We are now using *EventThresh* = 3.5, reserving an higher value (up to 7) for noisy stations only.

3.6 ClipCount (optional) and DeadSta

ClipCount specifies the maximum absolute amplitude (in counts zero-to-peak) that can be expected for a channel. *ClipCount* is not used by *pick_ew*, but by *eqcoda*, the sub-module that performs some coda envelop calculations, producing a coda duration and a coda weight for each arrival in a given event. ClipCount is not just related to the AD converter but it rather pertains to the entire chain sensor-(eventual modulator)- transmission line- (eventual demodulator)-AD converter, or (for a digital transmission) it depicts the characteristics of the couple sensor-AD converter in its linear response. As an example, an AD converter that is nominally at 24 bit, but that has a linear response up to 23.3 bits, should have a maximum (linear) output (and *ClipCount*) at +/-2^22.3 = +/- 5163793 counts. If that same AD converter had been set with a maximum input of +/-10V, but the associated sensor had a maximum output of +/-5 V, the *ClipCount* of the entire chain would be at 2^22 = 4194304 counts. The stage of the entire chain that has the minimum dynamic range limits the actual value of *ClipCount*.

DeadSta was used in the past to discover transmission interruptions. This parameter was of some utility when the analog transmission had a useful dynamic range that was lower than the dynamic range of the AD converter. In that case an abnormally high output of the AD converter could indicate a trouble in the channel, or a problem in the mass positioning. We suggest for analog transmission or with mass positioning systems:

DeadSta = 1.1 * *ClipCount* (in counts)

Some digitizers can introduce an offset in the output signal so that the actual maximum output of the chain sensor-digitizer has to be investigated carefully (although the high-pass pre-filtering should prevent this problem). To completely eliminate, where desired, the inhibitory effect of *DeadSta* we suggest to state:

DeadSta = 10 * *ClipCount* (in counts)

4. Event Termination/Evaluation Criteria (checking for noise)

4.1 MinPeakSize

MinPeakSize is the minimum size of the first three peaks of the (high-pass) filtered signal expressed in counts: after the declaration of a trigger at least one of the first three peaks has to be greater than *MinPeakSize*, otherwise the signal is declared as "noise". As a first choice in defining this parameter, we stated that the true (filtered) velocity of the ground has to exceed $3*10^{-8}$ m/s to declare a trigger. Hence we define, for each channel of a velocimeter:

$$MinPeakSize = V_{min} \; S_V$$

with

$$V_{min} = [\, 3 * 10^{-8} \text{ m/s} \,]$$

S_V is the sensitivity of the chain: velocimeter - digitizer, expressed in [counts/(m/s)]. Hence *MinPeakSize* is adimensional (a number of counts). We are testing *Pick_ew* also on some accelerometric signals; in this case we require a minimum acceleration of the ground. We suggest as starting definition of the parameter for an accelerometer:

$$MinPeakSize = A_{min} \; S_A$$

with

$A_{min} = [\ 3 * 10^{-4}\ m/s^2\ ]$

where S_A is the sensitivity of the chain: accelerometer → digitizer, expressed in [counts/(m/s^2)]. In some particular cases, in stations where the accelerometer is the only instrument installed, the site is not particularly noisy and the signal could be used to refine a location, we require:

$A_{min} = [\ 3 * 10^{-5}\ m/s^2\ ]$

4.2 MaxMint

MaxMint represents a time duration expressed in number of samples. Then it has to be computed using the sampling frequency of the stream. When the "event" has no zero crossings for *MaxMint* samples, the event is aborted. This parameter had an historical meaning, and was used in the first algorithm of Rex Allen to discharge possible false triggers caused by faults on the analog transmissions. We suggest to completely exclude the effect of this parameter for digital transmission channels stating:

$MaxMint = T_4 * sps = T_4 * (1/\Delta t)$

where sps is the sampling rate of the data stream and, for any instrument, $T_4 = 20$ s.

4.3 i9 (MinCodaLen in the program)

This is one of the few parameters of the *Pick_ew.sta* file that has already a physical dimension: it represents the minimum duration of a seismic signal in seconds. The picker algorithm refuses signals shorter than *MinCodaLen* seconds. In our experiments the triggering performance resulted to be very strongly affected by this parameter: the suggested value of 3 seconds caused in our case (*Italian National Seismic Network*) a large and unbearable number of false triggers. This fact might be attributed to very strong anthropic noise due to the high density of the population in Italy. For this same reason we have to keep *EventThresh* quite low and cannot reduce the number of false triggers using an higher STA/LTA value (that would result in a greater minimum magnitude of completeness for the entire network). We got much better results choosing 7 seconds as the minimum length of a seismic signal; a less populated area could allow shorter *MinCodaLen*. It might also be possible to choose a shorter minimum length for the majority of the stations of the seismic network, assigning a larger *MinCodaLen* only to a subset of very noisy stations.

MinCodaLen = 7 s (for velocimeter in noisy area).

For accelerometric instruments *MinCodaLen* has to be shortened to allow the detection of low magnitude events:

MinCodaLen = 3 s (for acceleromets).

4.4 Itr1, MinSmallZC and MinBigZC

While *Sta->isml* counts the small zero crossings (it is called "S" in the paper by Rex Allen, [1978]), and grows with the velocity of the counter of zero-crossings *Sta->m*, *itrm* (called "L" by Allen) grows with the velocity of *Sta->m / Itr1*. When S reaches L, the event is over.

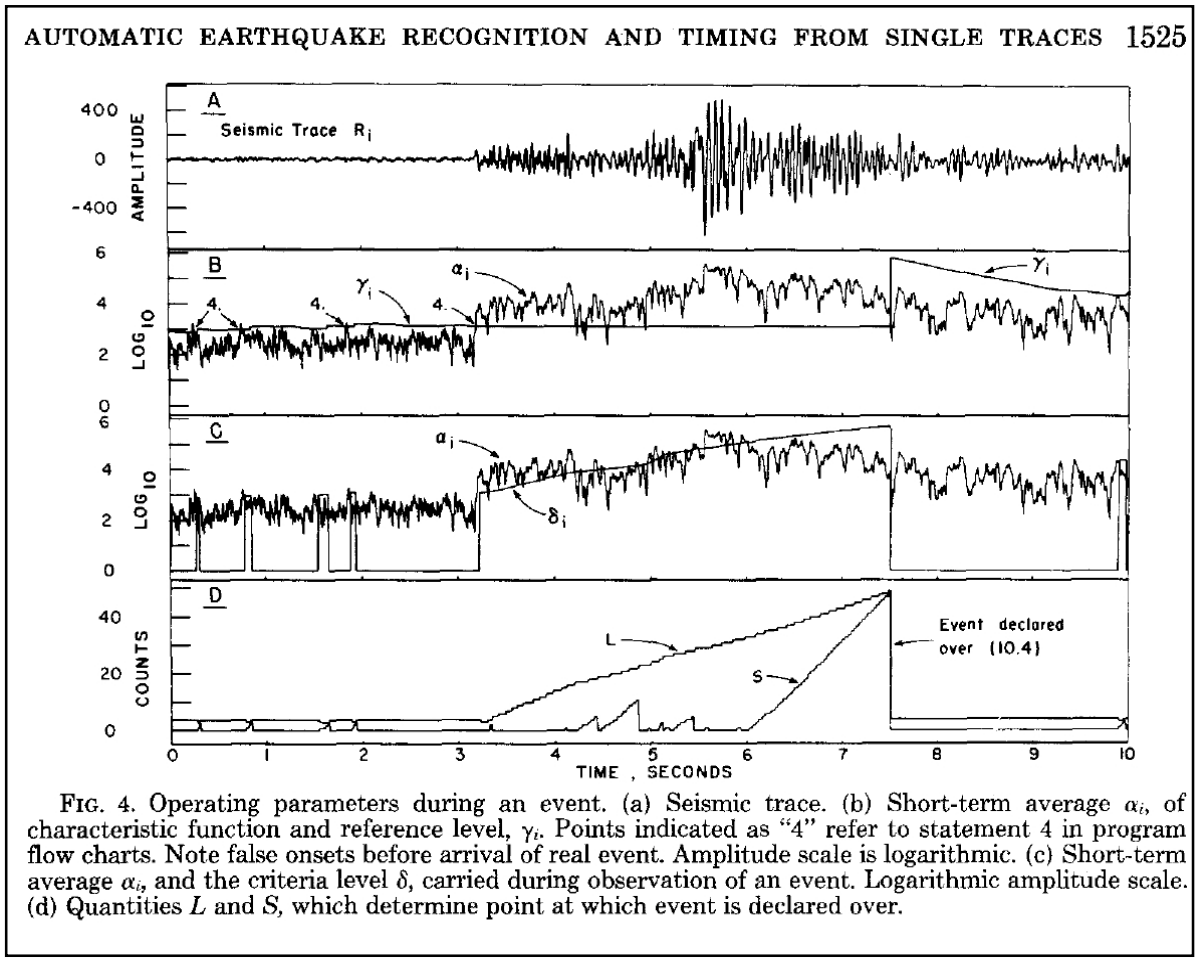

Figure 4. Reproduction of a figure taken from the paper of Allen (1978). The counters S and L in the picture are called *isml* and *itrm* respectively in the Earthworm realization of the algorithm. S counts the small zero crossings. Those are the zero crossings of the characteristic function, while its Short Term Average *esta* (α_i in the figure) remains over the threshold δ_i (see *ecrit* in paragraph 5.1). If *esta* goes under δ_i the counter S goes to 0. L is a trimming function whose derivative is 1/3 (=1/*Itr1*) of the derivative of S. When S reaches L the event is declared over. If this happens in a time shorter than *MiCodaLen*, the trigger is refused.

Itr1 is almost hard-wired in the code at 3 (although it is a parameter of *peak_ew*). The only statement where it is used follows (in *pick_ra.c*):

```
...omissis...
        /* Compute itrm, the number of small zero crossings to be allowed before declaring the pick
        over.  This will start out quite small and increase during the event to a maximum of 50.
        **********************************************/
                        itrm = Parm->Itr1 + (Sta->m / Parm->Itr1);
                        if ( Sta->m > 150 ) itrm = 50;
        /*  See if the pick is over
        ***********************/
                        if ( (++Sta->m != Parm->MinSmallZC) && (Sta->isml < itrm) )
                        continue;               /* It's not over */
...omissis...
```

The second statement presumes a ratio of 3 between the value of *Sta->m* (the total number of zero crossings from the beginning of the event) and the maximum value of *itrm* allowed. Greater values of *Itr1*

would make *itrm* grow too slowly, so that it might be reached by *isml* too quickly (a true trigger could be refused because its total duration might result short).

MinSmallZC is the number of zero-crossings (small and big) the algorithm waits before checking the number of big zero-crossings found. If, at that check, the number of big zero-crossings is at least *MinBigZC*, and at least one pick was bigger than *MinPeakSize*, then the signal is not noise. If *isml* reaches *itrm* before *MinSmallZC*, or the number of big zero-crossings is low, or no one of the maximum peaks reaches *MinPeakSize*, the signal is declared noise.

5. Coda Termination Parameters

5.1 Erefs

A trigger is declared if the short term average goes over the threshold *Sta->eref* (the threshold is called γ_i in Figure 4), computed by the long term average:

```
if ( Sta->esta    > Sta->eref )                    (scan.c)
```

but, after that declaration, for the following checks (used to declare the end of the event) a new threshold *ecrit* is computed (δ_i in Figure 4), that is slowly growing in this way:

```
old_eref   = Sta->eref;                            (scan.c)
Sta->ecrit  = old_eref                             (scan.c)
Sta->ecrit  += Sta->crtinc;                        (pick_ra-c)
```

where the last statement is applied at each zero-crossing, and

```
Sta->crtinc = Sta->eref / Parm->Erefs;             (scan.c)
```

The *Erefs* parameter can be interpreted as the number of zero-crossings necessary to double the reference level *ecrit* from its starting value *eref*. The reference level *ecrit* grows with time in order to ensure the end of a trigger also if it has been caused by a sudden and persistent cause (for instance a long lasting storm). It is clear, from it's definition, that *crtinc* is already scaled by the level of the noise and does not need further amplitude scaling. Moreover, the number of zero crossings does not dependent on the instrument sensitivity; it also has a weak relationship with the frequency response of the instrument (because the characteristic function itself is a low-pass filter of a signal that has been previously high-pass filtered). We suggest for all the channels *Erefs* = 50000.

5.2 CodaTerm, AltCoda and PreEvent

Pick_ew uses two methods to compute the coda length: which method is used depends on the noise level of the channel. Following the Earthworm documentation: "*For quiet stations, the coda is normally terminated when the absolute average of a 2-second waveform window drops below CodaTerm digital counts*".

The traditional definition of *CodaTerm* is "60 mV in counts". This definition was justified by a very homogeneous instrumentation in the network. If a seismic network exploits various instruments, it seems reasonable to scale *CodaTerm* using the sensitivity of each channel. We have already defined our usual triggering threshold *EventThresh* (=3.5); we have also defined a minimum peak size in terms of velocity as *MinPeakSize* = A_{min} * S_V = $3*10^{-8}$ m/s * S_V. With this choices, the maximum noise NS_{max} that permits the detection of the minimum seismic signal we want to record would be:

$$NS_{max} = 3 *10^{-8} \text{ (m/s)} \quad / \quad EventThresh = 0.857 *10^{-8} \text{ (m/s)}$$

To express the previous threshold in terms of counts we need to use the sensitivity:

$$CodaTerm = 0.857 *10^{-8} \text{ (m/s)} * S_V$$

where S_V is the sensitivity of the chain: velocimeter - digitizer, expressed in [counts/(m/s)]. Actually *CodaTerm* has to be thought of as the maximum noise allowed to detect the minimum signal we want to record, in counts. *Pick_ew* uses the 80% of *CodaTerm* to define a noisy station: that percentage is expressed in the parameter *AltCoda*, that has to be smaller then 1. When the average level of the noise is greater than *AltCoda* * *CodaTerm*, then the channel is declared noisy and the stopping threshold (for the computation of the duration of the signal) is defined multiplying the level of the noise by the factor *PreEvent* (usually 1.5). Remember that *PreEvent* has to be greater than 1, otherwise the length of the signal would be always equal to the maximum length allowed (144 seconds).

6. Vademecum

	Short period	5 s	BB, VBB
$StaFilt = 1 - e^{-2\pi \Delta t / T_1}$	T_1 = 0.068 s	T_1 = 0.126 s	T_1 = 0.123 s
$LtaFilt = 1 - e^{-2\pi \Delta t / T_2}$	T_2 = 2.06 s	T_2 = 3.14 s	T_2 = 4.19 s

	Velocimeter	Accelerometer
MinPeakSize = *Min.Signal * S*	*Min.Sign.* = $3*10^{-8}$ m/s *S*=Sensitivity in [counts/(m/s)]	*Min.Sign.* = $3*10^{-4}$ m/s^2 (if coupled with a velocimeter); *Min.Sign.* = $3*10^{-5}$ m/s^2 (if accelerometer alone); *S* = Sensitivity in [counts/ (m/s^2)]
MinCodaLen (i9)	7 s	3 s
$CodaTerm = NS_{max} * S$	NS_{max} = $0.857 * 10^{-8}$ (m/s) *S*=Sensitivity in [counts/(m/s)]	NS_{max} = $0.857 * 10^{-4}$ (m/s^2) *S*=Sensitivity in [counts/(m/s^2)]

	All instruments
$CharFuncFilt = C^*_2/\Delta t^2$	C^*_2 = 0.0003 s^2
$RmavFilt = e^{-2\pi \Delta t / T_3}$	T_3 = 16.08 s
$RawDataFilt = e^{-2\pi f_c \Delta t}$	f_c = 4.0 Hz
EventThresh	3.5 (most of the stations) 7 (noisy stations)
ClipCount	saturation of the chain sensor-digitizer in counts
DeadSta = 1.1 * *ClipCount*	(or *10 * ClipCount* to eliminate its effect)
MaxMint = T_4 (1/ Δt)	T_4 = 20 s
Itr1	3
MinSmallZC	50
MinBigZC	3
Erefs	50000
AltCoda	0.8
PreEvent	1.5

Acknowledgments

We would like to thank Dr. G. Romeo for useful discussions on electric and numerical filtering and an anonymous reviewer, whose suggestions helped to make this paper more readable.

References

Allen, R.V., (1978). Automatic earthquake recognition and timing from single traces. *Bull. Seism. Soc. Am*., vol. 68, no. 5, 1521-1532.

Allen, R.V., (1982). Automatic phase pickers: Their present use and future prospects. *Bull. Seism. Soc. Am*., vol.72, no.6, S225-S242.

Dixon, J.P., Power, J.A., Stihler, S.D., (2005). A Comparison of Seismic Event Detection with IASPEI and Earthworm Acquisition Systems at Alaskan Volcanoes. Seism. Res. Lett. 76 (2); 168-176; doi: 10.1785/gssrl.76.2.168

Gee, L.S., Neuhauser, D.S., Dreger, D.S., Pasyanos, M.E., Uhrhammer, R.A. and Romanowicz, B., (1996). Real-Time Seismology at UC Berkeley: The Rapid Earthquake Data Integration Project. Bull. Seism. Soc. Am., 86 (4), 936-945.

ISTI, (2004). Instrumental Software Technologies, Inc., http://www.isti.com/.

Johnson, C.E., Lindh, A. and Hirshorn, B., (1994). Robust regional phase association, U.S. Geol. Surv. Open-File Rept. 94-621.

Johnson, C.E., Bittenbinder, A., Bogaert, B., Dietz, L. and Kohler, W., (1995). Earthworm: A flexible approach to seismic network processing. IRIS Newsletter, 14 (2), 1-4.

Klein, F.W., (2002). User's guide to HYPOINVERSE-2000, a Fortran program to solve for earthquake locations and magnitudes, U.S. Geol. Surv. Open-File Rept. 02-0171, 123 pp.

Malone, S., (1998). Of Cathedrals, bazaars, and worms. Seism. Res. Lett., 69 (5).

Oppenheimer, D.H., Bittenbinder, A.N., Bogaert, B.M., Buland, R.P., Dietz, L.D., Hansen, R.A., Malone, S.D., McCreery, C.S., Sokolowski, T.J., Whitmore, P.M. and Weaver, C.S., (2005). The Seismic Project of the National Tsunami Hazard Mitigation Program, Nat. Hazards, 35, 59–72.

Pechmann, J.C., (1998-2006). Suggested Picker Parameter Changes (*Earthworm Documentation*), http.//www.isti2.com/ew/ovr/picker_tune.html.

USGS, CERI, ISTI and the Earthworm Community, (2010). Earthworm Documentation V7.4, http://folkworm.ceri.memphis.edu/ew-doc/

Wald, D.J., Quitoriano, V., Heaton, T.H., Kanamori, H., Scrivner, C.W. and Worden, C.B., (1999). TriNet ``ShakeMaps'': Rapid Generation of Instrumental Ground Motion and Intensity Maps for Earthquakes in Southern California. Earthquake Spectra, 15, 537-556.

APPENDIX A

This appendix reproduces part of the *pick_ew* documentation as found in:

Earthworm Modules: Pick_ew Configuration File Commands (last revised Mar 28, 2008)

in http://folkworm.ceri.memphis.edu/ew-doc/cmd/pick_ew_cmd.html

linked by:

Earthworm documentation V7.4, (April 27, 2010),

in http://folkworm.ceri.memphis.edu/ew-doc/

…omissis…

4. STATION LIST FORMAT

By default, pick_ew processes all TYPE_TRACEBUF messages (regardless of their installation id or module id) that reside on transport ring specified in the InRing command. All of the channels of trace data being processed by the Earthworm system should be described in pick_ew's station list file. This file contains one line per input channel. Each line contains 22 required fields used by pick_ew to identify the channel and set all the picking parameters for that channel. In v5.1 and higher an optional 23rd field was added for use by the module eqcoda. Pick_ew ignores this 23rd field if it exists. Upon retrieving a TYPE_TRACEBUF message from the ring, pick_ew finds the appropriate parameters for that message by matching the station, component, and network fields in the message header to a line from the station file. If it can't find a match, pick_ew won't process that tracebuf message.

A sample portion of a pick_ew station list file appears below.
Note that a comments are preceded by a #.

```
#
#  This is a sample station list for the pick_ew program.
#
#                                MinBigZC       RawDataFilt    LtaFilt         DeadSta          PreEvent
# Pick  Pin    Station/      MinSmallZC   MaxMint           StaFilt       RmavFilt          AltCoda
# Flag  Numb   Comp/Net/Loc    Itr1    MinPeakSize  i9  CharFuncFilt  EventThresh          CodaTerm         Erefs   ClipCount
# ----  ----   --------   ------------------------------------------------------------------------------------------------
    1     0  AAR   VHZ NC  -- 3  40  3  60  500  3  .985  3.  .6  .03  5.  .9961  1200.  49.14  .8  1.5  50000.     2048
    1     1  AAS   VHZ NC  -- 3  40  3  60  500  3  .985  3.  .6  .03  5.  .9961  1200.  49.14  .8  1.5  50000.     2048
    0     2  ABL   VHZ CI  -- 3  40  3  60  500  3  .985  3.  .6  .03  5.  .9961  1200.  49.14  .8  1.5  50000.     2048
    1     3  ABR   VHZ NC  -- 3  40  3  60  500  3  .985  3.  .6  .03  5.  .9961  1200.  49.14  .8  1.5  50000.     2048
    1  2005  KCPB  BHZ NC  -- 3  40  3 162  500  3  .939  3.  .4  .015 5.  .9961  1200. 132.7   .8  1.5 135000.  8388608
# end of station file
```

Each line in the station file contains the following 24 fields, separated by white space:

Channel Identification:

1. Pick Flag if Pick Flag is 0, pick_ew will not try to pick P-wave arrivals from this trace. If Pick Flag is 1, the trace will be picked.

2. Pin Numb this field is not used by pick_ew, but exists for posterity and for potential use by other programs. Each input signal in a given Earthworm system should have a pin number Pin Numb (2-byte integer) that is unique across all data sources within the system. For example, if we use two digitizers with 256 channels each, our pin numbers would range from 0 to 511.

3. Station — the first of 3 fields (Station-Comp-Net) that will uniquely identify each trace of seismic data. Station is a string, up to 5 characters, that identifies the physical site of the seismic instrument. This label must be unique within a given network.

4. Comp — a 3-character string to identify the component of motion recorded by this seismic trace.

5. Net — a 2-character string that identifies the network that operates the seismic instrument.

6. Loc — a 2-character string that identifies the location code that describes the sensor location.

Event Termination/Evaluation Criteria:

7. Itr1 — (i5 in earlier code) Sets Itr1 which is used to calculate the zero-crossing termination count. The pick_ew calculates itrm, the number of consecutive small-zero crossings (zero-crossings where the short-term average is less than the critical termination level) to be allowed before declaring the event over. itrm = Itr1 + m / Itr1 where m is the zero-crossing counter. itrm will start out quite small at the beginning of an event and will increase during an event to a maximum of 50.

8. MinSmallZC — (i6 in earlier code) Defines the minimum number of zero-crossings for a valid pick. An event is declared over and potentially valid after MinSmallZC zero-crossings. Pick_ew then evaluates the event to determine if it was a seismic event or noise.

9. MinBigZC — (i7 in earlier code) Defines the minimum number of "big zero-crossings" for a valid pick. No pick is reported unless at least MinBigZC big zero-crossings occurred while the event was active. [A "big zero-crossing" amplitude must exceed rbig, where rbig is (the largest amplitude of first 3 half-cycles after event activation)/3. A "big zero crossing" must also represent a crossing of opposite polarity to the previous crossing].

10. MinPeakSize — (i8 in earlier code) Defines the minimum amplitude (digital counts) for a valid pick. No pick is reported unless one of the first three peaks of an event has an amplitude larger than MinPeakSize digital counts.

11. MaxMint — (hard-wired at 500 in earlier code) Maximum interval (in samples) between zero crossings. If no zero crossings occur within MaxMint data samples, the pick event is terminated.

12. i9 — (i9 in earlier code) Defines the minimum coda length (seconds) for a valid pick. No pick is reported unless its coda is at least i9 seconds long.

Waveform Filtering Parameters:

13. RawDataFilt — (c1 in earlier code) Sets the filter parameter RawDataFilt that is applied to the raw trace data. This is essentially a recursive highpass filter that removes the DC offset from the data.

14. CharFuncFilt — (c2 in earlier code) Sets the filter parameter CharFuncFilt that is applied in the calculation of the characteristic function of the waveform data.

15. StaFilt (c3 in earlier code) Sets the filter parameter (time constant) StaFilt that is used in the calculation of the short-term average (STA) of the characteristic function of the trace.

16. LtaFilt (c4 in earlier code) Sets the filter parameter (time constant) LtaFilt that is used in the calculation of the long-term average (LTA) of the characteristic function of the trace.

17. EventThresh (c5 in earlier code) Sets the STA/LTA event threshold. An event (a pick) is declared when STA is larger than EventThresh*LTA. Once an event is declared, pick_ew enters an event evaluation mode to determine if the event is a P-wave arrival or noise.

18. RmavFilt Filter parameter (time constant) used to calculate the running mean of the absolute value of the waveform data. Usually set to 0.9961.

19. DeadSta (c6 in earlier code) Sets the dead station threshold (counts). If the running average of the absolute value (AAV) of a trace is greater than DeadSta digital counts, the channel is considered dead and the pick_ew does not attempt to pick arrivals.

Coda Termination Parameters:

20. CodaTerm (c7 in earlier code) Sets the "normal" coda termination threshold (counts). Two methods are used to determine if the coda length calculation is over. For quiet stations, the coda is "normally" terminated when the AAV of a 2-second waveform window drops below CodaTerm digital counts. For noisy stations, an alternate termination method is used (see AltCoda & PreEvent, below). Pick_ew stops all coda calculations 144 seconds after event activation, even if the coda AAV hasn't reached the termination value. In the Northern California Seismic Network's convention, CodaTerm is the number of digital counts produced by a discriminator output signal (input to the Earthworm A/D) of 60 mV. The 2-bit Earthworm A/D has an input range of +/- 2.5 volts and an output range of 0 to 4095 counts (-2048 to 2047 counts). For such a system, a 60 mV input will produce a sample of 49.1 counts.

21. AltCoda (c8 in earlier code) Defines the "noisy station level" at which pick_ew should use the alternate coda termination method. If the AAV of a channel is greater than CodaTerm*AltCoda digital counts when an event is declared, pick_ew uses the alternate coda termination method.

22. PreEvent (c9 in earlier code) Defines the alternate coda termination threshold for noisy stations. If the AAV of a channel is greater than CodaTerm*AltCoda digital counts when an event is declared, the coda is considered over when the AAV of a 2-second waveform window drops to less than PreEvent*(pre-event signal amplitude). Pick_ew stops all coda calculations 144 seconds after event activation, even if the coda AAV hasn't reached the termination value.

23. Erefs (hard-wired at 50000. in earlier code) Used in calculating the increment (crtinc) to be added to the criterion level (ecrit) at each zero crossing. The criterion level is used to determine if the event is over.

crtinc = eref / Erefs

where eref is the current STA/LTA reference level. The smaller Erefs is, the faster ecrit will increase, and the sooner the event will terminate.

Optional Parameters:

24. ClipCount (added in v5.1 for use by eqcoda, ignored by pick_ew) Specifies the maximum absolute amplitude (in counts zero-to-peak) that can be expected for this channel. Eqcoda calculates clipping thresholds for P-amplitudes and coda-window average absolute amplitudes as a fraction

Coordinamento editoriale e impaginazione
Centro Editoriale Nazionale | INGV

Progetto grafico e redazionale
Laboratorio Grafica e Immagini | INGV Roma

Via di Vigna Murata, 605
00143 Roma
Tel. +39 06518601 Fax +39 065041181

http://www.ingv.it

Istituto Nazionale di Geofisica e Vulcanologia