Working with EMA datafiles in Matlab

Jens Roeser & Adamantios Gafos

University of Potsdam
gafos@uni-potsdam.de

October 19, 2014

Table of Contents

1. Basics
2. EMA data structure
3. Signal processing
4. Velocity signals
5. Relating spatial and velocity signals
6. Storing labels

General outline for each section:

- Short introduction
- Try out solutions to proposed mini problems in your MATLAB console and write them into the provided MATLAB script
- Btw, open the provided MATLAB script

General outline for each section:

- Short introduction
- Try out solutions to proposed mini problems in your MATLAB console and write them into the provided MATLAB script
- Btw, open the provided MATLAB script

General outline for each section:

- Short introduction
- Try out solutions to proposed mini problems in your MATLAB console and write them into the provided MATLAB script
- Btw, open the provided MATLAB script

General outline for each section:

- Short introduction
- Try out solutions to proposed mini problems in your MATLAB console and write them into the provided MATLAB script
- Btw, open the provided MATLAB script

General outline for each section:

- Short introduction
- Try out solutions to proposed mini problems in your MATLAB console and write them into the provided MATLAB script
- Btw, open the provided MATLAB script

General outline for each section:

- Short introduction
- Try out solutions to proposed mini problems in your MATLAB console and write them into the provided MATLAB script
- Btw, open the provided MATLAB script

Table of Contents

1. Basics
2. EMA data structure
3. Signal processing
4. Velocity signals
5. Relating spatial and velocity signals
6. Storing labels

EMA data structure

- Go through sections 0. to 2. of the provided MATLAB script
- Basic syntax of MATLAB (0.)
- Setup work space (1.)
- Load data structures (2.)

EMA data structure

- Go through sections 0. to 2. of the provided MATLAB script
- Basic syntax of MATLAB (0.)
- Setup work space (1.)
- Load data structures (2.)

EMA data structure

- Go through sections 0. to 2. of the provided MATLAB script
- Basic syntax of MATLAB (0.)
- Setup work space (1.)
- Load data structures (2.)

EMA data structure

- Go through sections 0. to 2. of the provided MATLAB script
- Basic syntax of MATLAB (0.)
- Setup work space (1.)
- Load data structures (2.)

EMA data structure

- Go through sections 0. to 2. of the provided MATLAB script
- Basic syntax of MATLAB (0.)
- Setup work space (1.)
- Load data structures (2.)

Table of Contents

1. Basics
2. EMA data structure
3. Signal processing
4. Velocity signals
5. Relating spatial and velocity signals
6. Storing labels

Introduction

- EMA: spatial/ positional values recorded in electromagnetic articulography
- Acoustic information provided
- Files: data structure, i.e. structure arrays
- Structure arrays: collection of information of different types (e.g., strings, integers, floats) and lengths belonging to one object (here, stimulus)
- Fields: Subsets of the data contained in the structure array (i.e., receiver trajectory and audio wave form)

Introduction

- EMA: spatial/ positional values recorded in electromagnetic articulography
- Acoustic information provided
- Files: data structure, i.e. structure arrays
- Structure arrays: collection of information of different types (e.g., strings, integers, floats) and lengths belonging to one object (here, stimulus)
- Fields: Subsets of the data contained in the structure array (i.e., receiver trajectory and audio wave form)

Introduction

- EMA: spatial/ positional values recorded in electromagnetic articulography
- Acoustic information provided
- Files: data structure, i.e. structure arrays
- Structure arrays: collection of information of different types (e.g., strings, integers, floats) and lengths belonging to one object (here, stimulus)
- Fields: Subsets of the data contained in the structure array (i.e., receiver trajectory and audio wave form)

Introduction

- EMA: spatial/ positional values recorded in electromagnetic articulography
- Acoustic information provided
- Files: data structure, i.e. structure arrays
- Structure arrays: collection of information of different types (e.g., strings, integers, floats) and lengths belonging to one object (here, stimulus)
- Fields: Subsets of the data contained in the structure array (i.e., receiver trajectory and audio wave form)

Introduction

- EMA: spatial/ positional values recorded in electromagnetic articulography
- Acoustic information provided
- Files: data structure, i.e. structure arrays
- Structure arrays: collection of information of different types (e.g., strings, integers, floats) and lengths belonging to one object (here, stimulus)
- Fields: Subsets of the data contained in the structure array (i.e., receiver trajectory and audio wave form)

Introduction

- EMA: spatial/ positional values recorded in electromagnetic articulography
- Acoustic information provided
- Files: data structure, i.e. structure arrays
- Structure arrays: collection of information of different types (e.g., strings, integers, floats) and lengths belonging to one object (here, stimulus)
- Fields: Subsets of the data contained in the structure array (i.e., receiver trajectory and audio wave form)

EMA data structure

- Structure arrays contain different fields
- E.g. 'data' contains a 1×7 structure array
- fieldnames: SIGNAL, SRATE, NAME
- NAME: audio and 6 receiver trajectories
- SIGNAL and SRATE correspond to the different receivers in NAME

EMA data structure

- Structure arrays contain different fields
- E.g. 'data' contains a 1×7 structure array
- fieldnames: SIGNAL, SRATE, NAME
- NAME: audio and 6 receiver trajectories
- SIGNAL and SRATE correspond to the different receivers in NAME

EMA data structure

- Structure arrays contain different fields
- E.g. ‘data’ contains a 1×7 structure array
- fieldnames: SIGNAL, SRATE, NAME
- NAME: audio and 6 receiver trajectories
- SIGNAL and SRATE correspond to the different receivers in NAME

EMA data structure

- Structure arrays contain different fields
- E.g. ‘data’ contains a 1×7 structure array
- fieldnames: SIGNAL, SRATE, NAME
- NAME: audio and 6 receiver trajectories
- SIGNAL and SRATE correspond to the different receivers in NAME

EMA data structure

- Structure arrays contain different fields
- E.g. ‘data’ contains a $1{\times}7$ structure array
- fieldnames: SIGNAL, SRATE, NAME
- NAME: audio and 6 receiver trajectories
- SIGNAL and SRATE correspond to the different receivers in NAME

EMA data structure

- Structure arrays contain different fields
- E.g. ‘data’ contains a 1×7 structure array
- fieldnames: SIGNAL, SRATE, NAME
- NAME: audio and 6 receiver trajectories
- SIGNAL and SRATE correspond to the different receivers in NAME

Exercise 1

The field SRATE contains information about the sampling rate used during the data recording. As seen before for the file names and signal names, check which sampling rate was used for the field audio and for the individual trajectories (i.e., instead of asking for the NAME field, ask for the SRATE field). The order of the values for sampling rate corresponds to the names listed in ‘trajectories’.
Note: For now and later, provide the answers in your MATLAB script.

Table of Contents

1. Basics
2. EMA data structure
3. Signal processing
4. Velocity signals
5. Relating spatial and velocity signals
6. Storing labels

Signal processing

- SIGNAL: a sequence of values that are called samples spanning a time slice as determined by SRATE
- 2D data: contain positional signal for x- and y-dimension of tongue kinematics
- Samples are taken at regular time intervals as expressed in the sampling rate (in Hz).
- For a sampling rate of 400 Hz, 400 samples are taken per second or one sample all 0.0025 sec (i.e.,1/400)
- Proceed with **3. Signal processing** in the MATLAB script
- Afterwards, check Exercise 2 in the slides

Signal processing

- SIGNAL: a sequence of values that are called samples spanning a time slice as determined by SRATE
- 2D data: contain positional signal for x- and y-dimension of tongue kinematics
- Samples are taken at regular time intervals as expressed in the sampling rate (in Hz).
- For a sampling rate of 400 Hz, 400 samples are taken per second or one sample all 0.0025 sec (i.e.,1/400)
- Proceed with **3. Signal processing** in the MATLAB script
- Afterwards, check Exercise 2 in the slides

Signal processing

- SIGNAL: a sequence of values that are called samples spanning a time slice as determined by SRATE
- 2D data: contain positional signal for x- and y-dimension of tongue kinematics
- Samples are taken at regular time intervals as expressed in the sampling rate (in Hz).
- For a sampling rate of 400 Hz, 400 samples are taken per second or one sample all 0.0025 sec (i.e.,1/400)
- Proceed with **3. Signal processing** in the MATLAB script
- Afterwards, check Exercise 2 in the slides

Signal processing

- SIGNAL: a sequence of values that are called samples spanning a time slice as determined by SRATE
- 2D data: contain positional signal for x- and y-dimension of tongue kinematics
- Samples are taken at regular time intervals as expressed in the sampling rate (in Hz).
- For a sampling rate of 400 Hz, 400 samples are taken per second or one sample all 0.0025 sec (i.e.,1/400)
- Proceed with **3. Signal processing** in the MATLAB script
- Afterwards, check Exercise 2 in the slides

Signal processing

- SIGNAL: a sequence of values that are called samples spanning a time slice as determined by SRATE
- 2D data: contain positional signal for x- and y-dimension of tongue kinematics
- Samples are taken at regular time intervals as expressed in the sampling rate (in Hz).
- For a sampling rate of 400 Hz, 400 samples are taken per second or one sample all 0.0025 sec (i.e.,1/400)
- Proceed with **3. Signal processing** in the MATLAB script
- Afterwards, check Exercise 2 in the slides

Signal processing

- SIGNAL: a sequence of values that are called samples spanning a time slice as determined by SRATE
- 2D data: contain positional signal for x- and y-dimension of tongue kinematics
- Samples are taken at regular time intervals as expressed in the sampling rate (in Hz).
- For a sampling rate of 400 Hz, 400 samples are taken per second or one sample all 0.0025 sec (i.e.,1/400)
- Proceed with **3. Signal processing** in the MATLAB script
- Afterwards, check Exercise 2 in the slides

Signal processing

- SIGNAL: a sequence of values that are called samples spanning a time slice as determined by SRATE
- 2D data: contain positional signal for x- and y-dimension of tongue kinematics
- Samples are taken at regular time intervals as expressed in the sampling rate (in Hz).
- For a sampling rate of 400 Hz, 400 samples are taken per second or one sample all 0.0025 sec (i.e.,1/400)
- Proceed with **3. Signal processing** in the MATLAB script
- Afterwards, check Exercise 2 in the slides

Exercise 2

Repeat for the tongue tip trajectory (i.e., TTPOS) what you just learned for TBPOS. Use `strmatch` for TTPOS instead of TBPOS and index 'data'. Then, find the positional values x and y of the 266^{th} sample in the TTPOS trajectory.

Signal processing

- The duration of a signal can be retrieved from the amount of samples and the sampling rate.
- The following correspondence between time and samples is given:

$$f = 1/T \tag{1}$$

$$t = n/f \tag{2}$$

$$n = t^{*}f \tag{3}$$

Signal processing

- The duration of a signal can be retrieved from the amount of samples and the sampling rate.
- The following correspondence between time and samples is given:

$$f = 1/T \tag{1}$$

$$t = n/f \tag{2}$$

$$n = t^{*}f \tag{3}$$

Signal processing

- The duration of a signal can be retrieved from the amount of samples and the sampling rate.
- The following correspondence between time and samples is given:

$$f = 1/T \tag{1}$$

$$t = n/f \tag{2}$$

$$n = t^* f \tag{3}$$

Signal processing

- The duration of a signal can be retrieved from the amount of samples and the sampling rate.
- The following correspondence between time and samples is given:

$$f = 1/T \tag{1}$$

$$t = n/f \tag{2}$$

$$n = t^{*} f \tag{3}$$

Signal processing

- The duration of a signal can be retrieved from the amount of samples and the sampling rate.
- The following correspondence between time and samples is given:

$$f = 1/T \tag{1}$$

$$t = n/f \tag{2}$$

$$n = t^{*}f \tag{3}$$

Signal processing

- The duration of the signal in msec:

  `dur = 1000*(length(data(2).SIGNAL)-1)/data(2).SRATE`

- Sampling info starts from the first sample but time starts from 0.

Signal processing

- The duration of the signal in msec:

```
dur =
1000*(length(data(2).SIGNAL)-1)/data(2).SRATE
```

- Sampling info starts from the first sample but time starts from 0.

Signal processing

- The duration of the signal in msec:

  ```
  dur =
  1000*(length(data(2).SIGNAL)-1)/data(2).SRATE
  ```

- Sampling info starts from the first sample but time starts from 0.

Signal processing

- The duration of the signal in msec:

  ```
  dur =
  1000*(length(data(2).SIGNAL)-1)/data(2).SRATE
  ```

- Sampling info starts from the first sample but time starts from 0.

Exercise 3

- To work with different signals it is often useful to see how a (slice of a) signal looks like:
- Plot the y-signal of TBPOS: `plot(data(2).SIGNAL(:,2))`
- Create the same plot using the vector matrix of the TBPOS assigned before (e.g., `signal_tbp`).
- Plot the signal for the x-dimension of TTPOS by changing the relevant indices in the plot command. Remember, data.NAME gives the trajectory names for the index of TTPOS. The order of the trajectory names corresponds to the relevant index.
- Proceed with **4. Work on audio signals** in the MATLAB script

Exercise 3

- To work with different signals it is often useful to see how a (slice of a) signal looks like:
- Plot the y-signal of TBPOS: `plot(data(2).SIGNAL(:,2))`
- Create the same plot using the vector matrix of the TBPOS assigned before (e.g., `signal_tbp`).
- Plot the signal for the x-dimension of TTPOS by changing the relevant indices in the plot command. Remember, data.NAME gives the trajectory names for the index of TTPOS. The order of the trajectory names corresponds to the relevant index.
- Proceed with **4. Work on audio signals** in the MATLAB script

Exercise 3

- To work with different signals it is often useful to see how a (slice of a) signal looks like:
- Plot the y-signal of TBPOS: `plot(data(2).SIGNAL(:,2))`
- Create the same plot using the vector matrix of the TBPOS assigned before (e.g., `signal_tbp`).
- Plot the signal for the x-dimension of TTPOS by changing the relevant indices in the plot command. Remember, data.NAME gives the trajectory names for the index of TTPOS. The order of the trajectory names corresponds to the relevant index.
- Proceed with **4. Work on audio signals** in the MATLAB script

Exercise 3

- To work with different signals it is often useful to see how a (slice of a) signal looks like:
- Plot the y-signal of TBPOS: `plot(data(2).SIGNAL(:,2))`
- Create the same plot using the vector matrix of the TBPOS assigned before (e.g., `signal_tbp`).
- Plot the signal for the x-dimension of TTPOS by changing the relevant indices in the plot command. Remember, data.NAME gives the trajectory names for the index of TTPOS. The order of the trajectory names corresponds to the relevant index.
- Proceed with **4. Work on audio signals** in the MATLAB script

Exercise 3

- To work with different signals it is often useful to see how a (slice of a) signal looks like:
- Plot the y-signal of TBPOS: `plot(data(2).SIGNAL(:,2))`
- Create the same plot using the vector matrix of the TBPOS assigned before (e.g., `signal_tbp`).
- Plot the signal for the x-dimension of TTPOS by changing the relevant indices in the plot command. Remember, data.NAME gives the trajectory names for the index of TTPOS. The order of the trajectory names corresponds to the relevant index.
- Proceed with **4. Work on audio signals** in the MATLAB script

Exercise 3

- To work with different signals it is often useful to see how a (slice of a) signal looks like:
- Plot the y-signal of TBPOS: `plot(data(2).SIGNAL(:,2))`
- Create the same plot using the vector matrix of the TBPOS assigned before (e.g., `signal_tbp`).
- Plot the signal for the x-dimension of TTPOS by changing the relevant indices in the plot command. Remember, data.NAME gives the trajectory names for the index of TTPOS. The order of the trajectory names corresponds to the relevant index.
- Proceed with **4. Work on audio signals** in the MATLAB script

Exercise 4

MATLAB might give you a warning when you extract the audio signal. This can be circumvented by normalizing the amplitude of the signal to 1 as shown in the code below. The normalized audio signal is assigned to `audio_signal`. Normalize the audio signal before saving it to a .wav file. Apply the function `audiowrite` to `audio_signal`. No warning should appear in the console if previously seen.

```
audio_signal =
data(1).SIGNAL/max(abs(data(1).SIGNAL));
```

Exercise 4

MATLAB might give you a warning when you extract the audio signal. This can be circumvented by normalizing the amplitude of the signal to 1 as shown in the code below. The normalized audio signal is assigned to `audio_signal`. Normalize the audio signal before saving it to a .wav file. Apply the function `audiowrite` to `audio_signal`. No warning should appear in the console if previously seen.

```
audio_signal =
data(1).SIGNAL/max(abs(data(1).SIGNAL));
```

Signal processing

- All signals in the structure array are synchronous.
- The n^{th} sample has positional representations in all receiver trajectories.
- Try, `data(t).SIGNAL(552,:)` and replace `t` for different receivers.
- Display spatial values of vertical movements for the 432^{nd} to 440^{th} sample of JAWPOS (i.e., jaw posture) and LLPOS (i.e., lower lip posture)
- Proceed with **5. Local extrema** in the MATLAB script

Signal processing

- All signals in the structure array are synchronous.
- The n^{th} sample has positional representations in all receiver trajectories.
- Try, `data(t).SIGNAL(552,:)` and replace `t` for different receivers.
- Display spatial values of vertical movements for the 432^{nd} to 440^{th} sample of JAWPOS (i.e., jaw posture) and LLPOS (i.e., lower lip posture)
- Proceed with **5. Local extrema** in the MATLAB script

Signal processing

- All signals in the structure array are synchronous.
- The n^{th} sample has positional representations in all receiver trajectories.
- Try, `data(t).SIGNAL(552,:)` and replace `t` for different receivers.
- Display spatial values of vertical movements for the 432^{nd} to 440^{th} sample of JAWPOS (i.e., jaw posture) and LLPOS (i.e., lower lip posture)
- Proceed with **5. Local extrema** in the MATLAB script

Signal processing

- All signals in the structure array are synchronous.
- The n^{th} sample has positional representations in all receiver trajectories.
- Try, `data(t).SIGNAL(552,:)` and replace `t` for different receivers.
- Display spatial values of vertical movements for the 432^{nd} to 440^{th} sample of JAWPOS (i.e., jaw posture) and LLPOS (i.e., lower lip posture)
- Proceed with **5. Local extrema** in the MATLAB script

Signal processing

- All signals in the structure array are synchronous.
- The n^{th} sample has positional representations in all receiver trajectories.
- Try, `data(t).SIGNAL(552,:)` and replace `t` for different receivers.
- Display spatial values of vertical movements for the 432^{nd} to 440^{th} sample of JAWPOS (i.e., jaw posture) and LLPOS (i.e., lower lip posture)
- Proceed with **5. Local extrema** in the MATLAB script

Signal processing

- All signals in the structure array are synchronous.
- The n^{th} sample has positional representations in all receiver trajectories.
- Try, `data(t).SIGNAL(552,:)` and replace `t` for different receivers.
- Display spatial values of vertical movements for the 432^{nd} to 440^{th} sample of JAWPOS (i.e., jaw posture) and LLPOS (i.e., lower lip posture)
- Proceed with **5. Local extrema** in the MATLAB script

Table of Contents

1. Basics
2. EMA data structure
3. Signal processing
4. Velocity signals
5. Relating spatial and velocity signals
6. Storing labels

Velocity signals

- Velocity corresponds to the acceleration of a gesture.
- The maximum in velocity can be found at the onset of a gestural movement and when the offset of a gesture.
- The minimum in velocity is the maximum constriction of a gesture.

Velocity signals

- Velocity corresponds to the acceleration of a gesture.
- The maximum in velocity can be found at the onset of a gestural movement and when the offset of a gesture.
- The minimum in velocity is the maximum constriction of a gesture.

Velocity signals

- Velocity corresponds to the acceleration of a gesture.
- The maximum in velocity can be found at the onset of a gestural movement and when the offset of a gesture.
- The minimum in velocity is the maximum constriction of a gesture.

Velocity signals

- Velocity corresponds to the acceleration of a gesture.
- The maximum in velocity can be found at the onset of a gestural movement and when the offset of a gesture.
- The minimum in velocity is the maximum constriction of a gesture.

Velocity signals

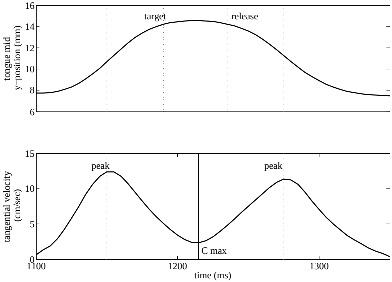

Figure: The positional signal in the vertical dimension (upper panel) and the magnitude of the tangential velocity of the tongue mid receiver (lower panel) plotted as a function of time.

Velocity signals

- See **5. Velocity signals** in the MATLAB script
- The velocity depends on the signal of a receiver (here, TBPOS is used.
- `s = data(2).SIGNAL;`
- *Central difference approximation* calculates the derivative of the dimensional velocities on the basis of samples:

```
vel = [diff(s(1:2,:))  ; s(3:end,:)  -
s(1:end-2,:)  ; diff(s(end-1:end,:))]  ./ 2;
```

Velocity signals

- See **5. Velocity signals** in the MATLAB script
- The velocity depends on the signal of a receiver (here, TBPOS is used.
- `s = data(2).SIGNAL;`
- *Central difference approximation* calculates the derivative of the dimensional velocities on the basis of samples:

```
vel = [diff(s(1:2,:))  ; s(3:end,:)  -
s(1:end-2,:)  ; diff(s(end-1:end,:))]  ./ 2;
```

Velocity signals

- See **5. Velocity signals** in the MATLAB script
- The velocity depends on the signal of a receiver (here, TBPOS is used.
- `s = data(2).SIGNAL;`
- *Central difference approximation* calculates the derivative of the dimensional velocities on the basis of samples:

```
vel = [diff(s(1:2,:))  ; s(3:end,:)  -
s(1:end-2,:)  ; diff(s(end-1:end,:))]  ./ 2;
```

Velocity signals

- See **5. Velocity signals** in the MATLAB script
- The velocity depends on the signal of a receiver (here, TBPOS is used.
- `s = data(2).SIGNAL;`
- *Central difference approximation* calculates the derivative of the dimensional velocities on the basis of samples:

```
vel = [diff(s(1:2,:))  ; s(3:end,:)  -
s(1:end-2,:)  ; diff(s(end-1:end,:))]  ./ 2;
```

Velocity signals

- See **5. Velocity signals** in the MATLAB script
- The velocity depends on the signal of a receiver (here, TBPOS is used.
- `s = data(2).SIGNAL;`
- *Central difference approximation* calculates the derivative of the dimensional velocities on the basis of samples:

```
vel = [diff(s(1:2,:))  ; s(3:end,:)  -
s(1:end-2,:)  ; diff(s(end-1:end,:))]  ./ 2;
```

Velocity signals

- See **5. Velocity signals** in the MATLAB script
- The velocity depends on the signal of a receiver (here, TBPOS is used.
- `s = data(2).SIGNAL;`
- *Central difference approximation* calculates the derivative of the dimensional velocities on the basis of samples:

```
vel = [diff(s(1:2,:))  ; s(3:end,:)  -
s(1:end-2,:)  ; diff(s(end-1:end,:))]  ./ 2;
```

Velocity signals

- Individual dimension velocities in cm/sec are calculated from `vel` by indexing x and y dimension.
- Multiply sampling rate and signal of the respective dimension (samples to sec) and from mm to cm and append results to new field in data structure

  `data(2).VEL_y = data(2).SRATE*vel(:,2)./10;` % for the vertical velocity in y dimension

  `data(2).VEL_y = data(2).SRATE*vel(:,1)./10;` % for the horizontal velocity in x dimension

Velocity signals

- Individual dimension velocities in cm/sec are calculated from `vel` by indexing x and y dimension.
- Multiply sampling rate and signal of the respective dimension (samples to sec) and from mm to cm and append results to new field in data structure

  `data(2).VEL_y = data(2).SRATE*vel(:,2)./10;` % for the vertical velocity in y dimension

  `data(2).VEL_y = data(2).SRATE*vel(:,1)./10;` % for the horizontal velocity in x dimension

Velocity signals

- Individual dimension velocities in cm/sec are calculated from `vel` by indexing x and y dimension.
- Multiply sampling rate and signal of the respective dimension (samples to sec) and from mm to cm and append results to new field in data structure

  `data(2).VEL_y = data(2).SRATE*vel(:,2)./10;` % for the vertical velocity in y dimension

  `data(2).VEL_y = data(2).SRATE*vel(:,1)./10;` % for the horizontal velocity in x dimension

Velocity signals

- Individual dimension velocities in cm/sec are calculated from `vel` by indexing x and y dimension.
- Multiply sampling rate and signal of the respective dimension (samples to sec) and from mm to cm and append results to new field in data structure

  `data(2).VEL_y = data(2).SRATE*vel(:,2)./10; % for the vertical velocity in y dimension`

  `data(2).VEL_y = data(2).SRATE*vel(:,1)./10; % for the horizontal velocity in x dimension`

Velocity signals

- Individual dimension velocities in cm/sec are calculated from `vel` by indexing x and y dimension.
- Multiply sampling rate and signal of the respective dimension (samples to sec) and from mm to cm and append results to new field in data structure

  `data(2).VEL_y = data(2).SRATE*vel(:,2)./10;` % for the vertical velocity in y dimension

  `data(2).VEL_y = data(2).SRATE*vel(:,1)./10;` % for the horizontal velocity in x dimension

Velocity signals

- Tangential velocity v_t is defined as the square root of the sum of the squared dimensional velocities:

$$v_t = \sqrt{v_x^2 + v_y^2} \tag{4}$$

```
v_t = sqrt(sum(vel.^2,2));
data(2).VEL_tang = data(2).SRATE*v_t./10;
```

- Check data(2)

Velocity signals

- Tangential velocity v_t is defined as the square root of the sum of the squared dimensional velocities:

$$v_t = \sqrt{v_x^2 + v_y^2} \tag{4}$$

```
v_t = sqrt(sum(vel.^2,2));
data(2).VEL_tang = data(2).SRATE*v_t./10;
```

- Check `data(2)`

Velocity signals

- Tangential velocity v_t is defined as the square root of the sum of the squared dimensional velocities:

$$v_t = \sqrt{v_x^2 + v_y^2} \tag{4}$$

```
v_t = sqrt(sum(vel.^2,2));
data(2).VEL_tang = data(2).SRATE*v_t./10;
```

- Check data(2)

Velocity signals

- Tangential velocity v_t is defined as the square root of the sum of the squared dimensional velocities:

$$v_t = \sqrt{v_x^2 + v_y^2} \tag{4}$$

```
v_t = sqrt(sum(vel.^2,2));
data(2).VEL_tang = data(2).SRATE*v_t./10;
```

- Check data(2)

Velocity signals

- Tangential velocity v_t is defined as the square root of the sum of the squared dimensional velocities:

$$v_t = \sqrt{v_x^2 + v_y^2} \tag{4}$$

```
v_t = sqrt(sum(vel.^2,2));
data(2).VEL_tang = data(2).SRATE*v_t./10;
```

- Check `data(2)`

Velocity signals

- Tangential velocity v_t is defined as the square root of the sum of the squared dimensional velocities:

$$v_t = \sqrt{v_x^2 + v_y^2} \tag{4}$$

```
v_t = sqrt(sum(vel.^2,2));
data(2).VEL_tang = data(2).SRATE*v_t./10;
```

- Check `data(2)`

Exercise 5

Plot the tangential velocity of TBPOS in the region (`rng`) that has been used for the local maximum (e.g., 250 to 270). Also, determine the corresponding local minimum of the tangential velocity given a time stamp at 650 ms. This minimum corresponds to what labeling software of articulatory kinematics considers to be a maximum constriction. You basically will have to change the field SIGNAL to `VEL_tang` in the exercise **5. Local extrema**. Then, calculate the minimum by applying the function min instead of max (use the help function for min). You might see that the given range does not include a local minimum. In this case, increase the range and re-plot. If you follow the detection of the local maximum, this exercise can be solved easily. What is the time stamp of the maximum constriction?

Table of Contents

1. Basics
2. EMA data structure
3. Signal processing
4. Velocity signals
5. Relating spatial and velocity signals
6. Storing labels

Relating spatial and velocity signals

- Different receivers have corresponding spatial signals at the each sample index.
- Gestural movements can be expressed spatially and in terms of dimensional or tangential velocity.
- Use the `subplot` function to illustrate the correspondence between spatial and velocity signals. See **7. Relating spatial and velocity signals** in the MATLAB script.
- How is the spatial signal related to the velocity signals?

Relating spatial and velocity signals

- Different receivers have corresponding spatial signals at the each sample index.
- Gestural movements can be expressed spatially and in terms of dimensional or tangential velocity.
- Use the `subplot` function to illustrate the correspondence between spatial and velocity signals. See **7. Relating spatial and velocity signals** in the MATLAB script.
- How is the spatial signal related to the velocity signals?

Relating spatial and velocity signals

- Different receivers have corresponding spatial signals at the each sample index.
- Gestural movements can be expressed spatially and in terms of dimensional or tangential velocity.
- Use the `subplot` function to illustrate the correspondence between spatial and velocity signals. See **7. Relating spatial and velocity signals** in the MATLAB script.
- How is the spatial signal related to the velocity signals?

Relating spatial and velocity signals

- Different receivers have corresponding spatial signals at the each sample index.
- Gestural movements can be expressed spatially and in terms of dimensional or tangential velocity.
- Use the `subplot` function to illustrate the correspondence between spatial and velocity signals. See **7. Relating spatial and velocity signals** in the MATLAB script.
- How is the spatial signal related to the velocity signals?

Relating spatial and velocity signals

- Different receivers have corresponding spatial signals at the each sample index.
- Gestural movements can be expressed spatially and in terms of dimensional or tangential velocity.
- Use the `subplot` function to illustrate the correspondence between spatial and velocity signals. See **7. Relating spatial and velocity signals** in the MATLAB script.
- How is the spatial signal related to the velocity signals?

Exercise 6

Create similar plots for TMPOS. Observe the sensitivity to local extrema under the variation of the sample range. Vary the sample range between e.g. 200 to 700, 300 to 500, 400 to 450). You will have to specify `s` and calculate `vel` for `s` of TMPOS. Create a new vector `rng` that contains the sample range values.

When done, continue with the MATLAB script **8. Plotting II**.

Exercise 6

Create similar plots for TMPOS. Observe the sensitivity to local extrema under the variation of the sample range. Vary the sample range between e.g. 200 to 700, 300 to 500, 400 to 450). You will have to specify `s` and calculate `vel` for `s` of TMPOS. Create a new vector `rng` that contains the sample range values.

When done, continue with the MATLAB script **8. Plotting II**.

Exercise 6

Create similar plots for TMPOS. Observe the sensitivity to local extrema under the variation of the sample range. Vary the sample range between e.g. 200 to 700, 300 to 500, 400 to 450). You will have to specify `s` and calculate `vel` for `s` of TMPOS. Create a new vector `rng` that contains the sample range values.

When done, continue with the MATLAB script **8. Plotting II**.

Table of Contents

1. Basics
2. EMA data structure
3. Signal processing
4. Velocity signals
5. Relating spatial and velocity signals
6. Storing labels

Storing labels

- Extracted information can be saved in a new structure array, e.g. `labels`
- The first entry gets the index 1, `labels(1)`
- Use the dot operator to add fields, e.g. `labels(1).file`
- Continue with the MATLAB script **9. Storing labels**

Storing labels

- Extracted information can be saved in a new structure array, e.g. `labels`
- The first entry gets the index 1, `labels(1)`
- Use the dot operator to add fields, e.g. `labels(1).file`
- Continue with the MATLAB script **9. Storing labels**

Storing labels

- Extracted information can be saved in a new structure array, e.g. `labels`
- The first entry gets the index 1, `labels(1)`
- Use the dot operator to add fields, e.g. `labels(1).file`
- Continue with the MATLAB script **9. Storing labels**

Storing labels

- Extracted information can be saved in a new structure array, e.g. `labels`
- The first entry gets the index 1, `labels(1)`
- Use the dot operator to add fields, e.g. `labels(1).file`
- Continue with the MATLAB script **9. Storing labels**

Storing labels

- Extracted information can be saved in a new structure array, e.g. `labels`
- The first entry gets the index 1, `labels(1)`
- Use the dot operator to add fields, e.g. `labels(1).file`
- Continue with the MATLAB script **9. Storing labels**

That's all ;)