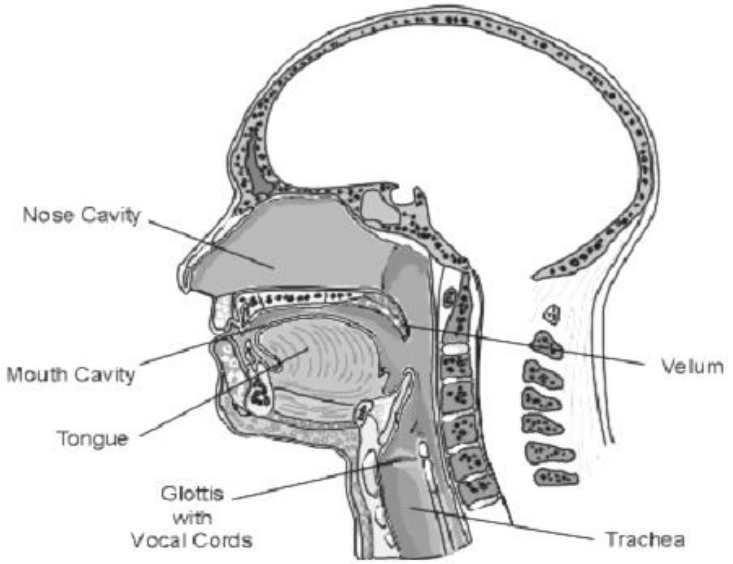
BAB 2
LANDASAN TEORI

2.1 Teori Umum Suara

Suara adalah fenomena fisik yang dihasilkan oleh getaran suatu benda yang berdupa sinyal analog dengan amplitudo yang berubah secara kontinyu terhadap waktu. Suara berhubungan erat dengan rasa "mendengar". Suara atau bunyi biasanya merambat melalui udara. Suara atau bunyi tidak bisa merambat melalui ruang hampa [**1**].

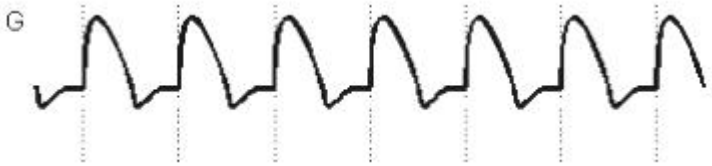
2.2 Proses Produksi Suara

Speech (wicara) dihasilkan dari sebuah kerjasama antara *lungs* (paru-paru), *glottis* (dengan *vocal cords*) dan *articulation tract* (*mouth*/mulut dan *nose cavity*/rongga hidung). Gambar 2.1 menunjukan penampang melintang dari organ wicara manusia. Untuk menghasilkan sebuah *voiced sounds* (suara ucapan), paru-paru menekan udara melalui *epiglottis*, *vocal cords* bergetar, menginterupt udara melalui aliran udara dan menghasilkan sebuah gelombang tekanan *quasi-periodic* [**2**].

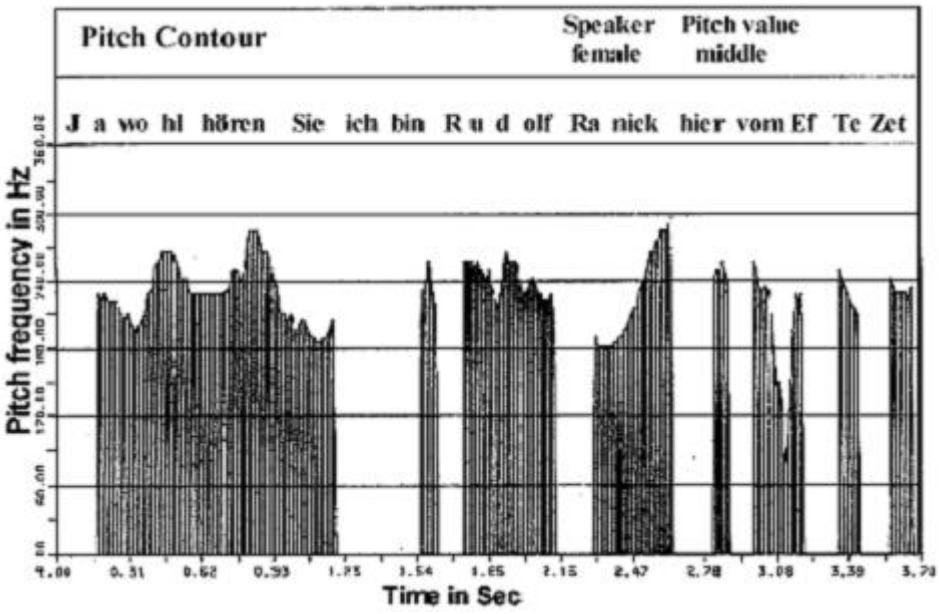


Gambar 2.1 Organ Wicara Manusia

Impuls tekanan pada umumnya disebut sebagai *pitch impulses* dan frekuensi sinyal tekanan adalah *pitch frequency* atau *fundamental frequency*. Pada Gambar 2.2 sederetan impuls (fungsi tekanan suara) dihasikan oleh *vocal cords* untuk sebuah suara. Ini merupakan bagian dari sinyal *voice* (suara) yang mendefinisikan *speech melody* (melodi wicara). Ketika seseorang berbicara dengan sebuah frekuensi *pitch* konstan, suara sinyal wicara *monotonous* tetapi dalam kasus normal sebuah perubahan permanen pada frekuensi terjadi. Ketika seseorang berbicara dengan frekuensi *pitch constant*, maka impuls *pitch* akan terlihat seperti pada Gambar 2.2. Variasi frekuensi *pitch* dapat dilihat seperti yang terlihat pada Gambar 2.3 di bawah ini.



Gambar 2.2 Sederet Impuls yang sama

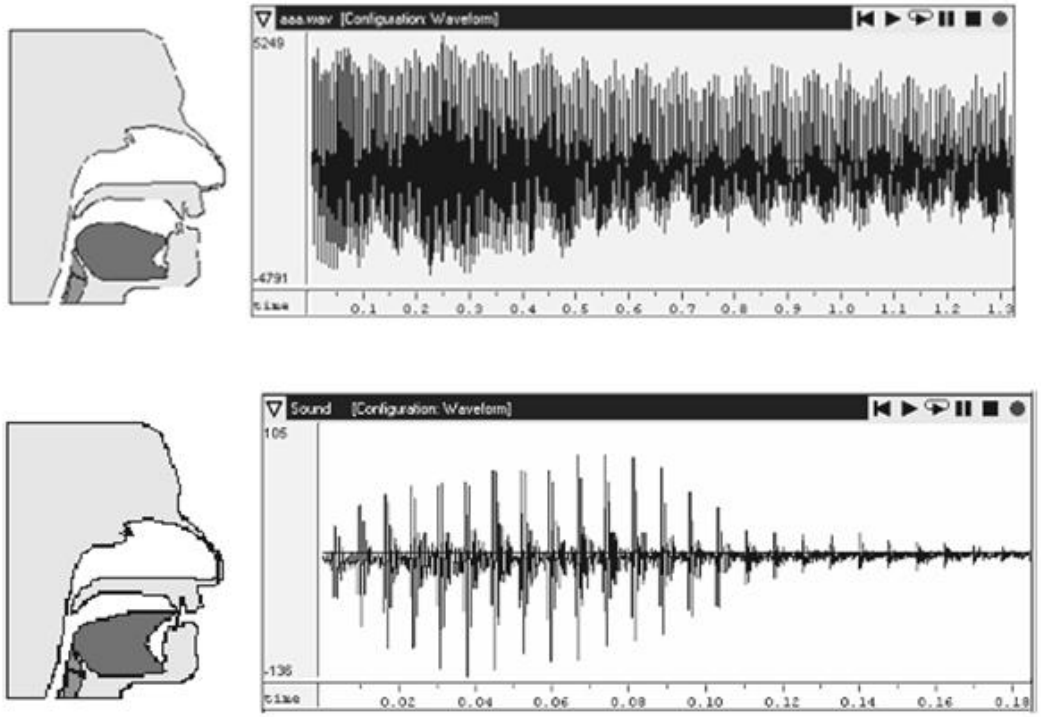


Gambar 2.3 Variasi Frekuensi

Impuls *pitch* merangsang udara di dalam mulut, dan untuk suara tertentu (*nasals*) juga merangsang *nasal cavity* (rongga hidung). Ketika rongga beresonansi, akan menimbulkan radiasi sebuah gelombang suara yang mana

merupakan sinyal wicara. Kedua rongga beraksi sebagai resonators dengan karakteristik frekuensi resonansi masing-masing, yang disebut *formant frequencies*. Pada saat rongga mulut dapat mengalami perubahan besar, manusia mampu untuk menghasilkan beragam pola ucapan suara yang berbeda. Mampu pula untuk menghasilkan beragam pola ucapan suara yang berbeda.

Di dalam kasus *unvoiced sounds* (suara tak terucap), eksitasi pada *vocal tract* lebih menyerupai *noise* (derau). Gambar 2.4 menampilkan proses produksi suara-suara "*a*", dan "*f*". Untuk sementara perbedaan bentuk dan posisi pada organ artikulasi diabaikan saja.



Gambar 2.4 Proses Produksi Suara Ucapan 'a' dan Ucapan 'f'

Berbagai informasi yang dapat diperoleh dari suara yang berasal dari mulut manusia dapat dilihat pada Tabel 2.1 di bawah ini.

Tabel 2.1 Informasi Suara

No	Informasi Suara
1	Identitas pengucap
2	Jenis kelamin

No	Informasi Suara
3	Usia
4	Tingkat Kejenuhan
5	Emosi
6	Jarak
7	Pengenalan kata
8	Kondisi kesehatan
9	Logat
10	Laju suara
11	Tingkat kebisingan
12	Kualitas bahasa

Setiap ucapan yang dikeluarkan dari mulut manusia akan mengandung berbagai informasi seperti pada Tabel 2.2, baik itu disadari maupun tidak. Misalkan Wisnu mengucapkan kata “diam” kepada Arif dengan nada keras, maka kata “diam” tersebut dapat diklasifikasikan seperti dalam Tabel 2.2 [**2**].

Tabel 2.2 Jenis Klasifikasi Suara

No	Informasi	Jenis Klasifikasi/Identifikasi
1	Identitas pengucap	Wisnu
2	Jenis gender	Pria
3	Usia	20-25 tahun
4	Ekspresi/emosi	Marah
5	Jarak sumber suara	± 5 meter
6	Pengenalan kata	Diam
7	Dialek/logat/suku	Jawa
8	Laju suara	Cepat
9	Tingkat kebisingan	0,8 (interval 0-1)
10	Tingkat kejenuhan	0,6 (interval 0-1)
11	Kualitas bahasa	Terdidik, pasaran, normal
12	Kondisi kesehatan	Normal

2.2.1 Jenis Suara Manusia

Manusia memiliki ciri khas suara yang berbeda-beda. Ciri khas tersebut dibedakan melalui nilai fitur *pitch* dan *formant*. Jenis-jenis suara manusia ada

yang tinggi, sedang, dan rendah. Dalam skripsi ini dilakukan penelitian hanya untuk suara tinggi dan rendah yang diantaranya suara pria tinggi (tenor), suara pria rendah (bass), suara wanita tinggi (sopran), dan suara wanita rendah (alto) [**19**]. Berikut dijelaskan dalam Tabel 2.3 di bawah ini.

Tabel 2.3 Jenis Suara Manusia

Pria		**Wanita**	
Tenor (tinggi)	Bass (rendah)	Sopran (tinggi)	Alto (rendah)
130.81 Hz - 523.25 Hz	82.407 Hz - 329.63 Hz	261.63 Hz - 1046.5 Hz	174.61 Hz - 698.46 Hz

2.2.2 Pitch

Pitch merupakan frekuensi fundamental (F_0) dari sinyal suara yang merupakan hasil akustik kecepatan getaran pita suara. Semakin besar getaran pita suara, maka akan semakin tinggi nilai *pitch*. Periode *pitch* berkisar antara 10 sampai 20 milidetik. Setiap manusia mempunyai kisaran *pitch* tersendiri, tergantung dari pangkal tenggorok yang dimiliki. Kisaran *pitch* yang khas (*habitual pitch*) dimiliki oleh kebanyakan pria sebesar 50 Hz – 250 Hz, sedangkan wanita memiliki *pitch* (*habitual pitch*) yang lebih tinggi dibandingkan dengan pria, yaitu berkisar antara 120 Hz – 500 Hz. Frekuensi fundamental ini berubah secara konstan dan memberikan informasi *linguistic* seseorang seperti pembeda antara intonasi dan emosi.

Pada pria ketika bersuara trakea dan laring pada tenggorokan membuka lebih lebar dibandingkan pada wanita. Ukuran pita suara pada pria berkisar antara 17.5 mm sampai 25 mm, sedangkan pada wanita ukuran pita suaranya berkisar antara 12.5 sampai 17.5 mm. Karena ukuran pita suara wanita lebih kecil, maka suara yang dihasilkan oleh wanita akan lebih tinggi [**1**][**7**].

2.2.3 Formant

Formant adalah frekuensi resonansi alami yang terjadi didalam rongga bidang suara, tergantung pada bentuk dan ukuran bidang suara dan lebih

menyerupai gaung [**1**]. Frekuensi *formant* bersifat tidak terbatas namun, untuk mengidentifikasi seseorang paling tidak ada 3 (tiga) format yang dianalisa, yaitu *formant* 1 (F_1), *formant* 2 (F_2), dan *formant* 3 (F_3) [**3**]. *Formant* pertama dan *formant* kedua berkaitan dengan posisi lidah ketika berbicara, sedangkan *formant* ketiga berpengaruh terhadap warna suara yang dihasilkan. Nilai *formant* pertama (F_1) berkaitan dengan posisi lidah terhadap langit-langit rongga mulut. Semakin dekat lidah dengan langit-langit, maka frekuensi yang dihasilkan semakin kecil. Nilai *formant* kedua (F_2) berkaitan dengan posisi lidah depan dan belakang. Nilai *formant* (F_2) yang tinggi dihasilkan ketika posisi lidah berada di depan, jika posisi lidah berada di belakang, maka nilai *formant* kedua (F_2) akan lebih kecil dibandingkan dengan yang posisi lidah di depan.

2.3 Kecerdasan Buatan

Kecerdasan buatan atau *Artificial Intelligence* (AI) merupakan cabang ilmu yang berusaha memahami kecerdasan manusia. AI berusaha membangun entitas-entitas cerdas yang sesuai dengan pemahaman manusia. Entitas-entitas cerdas yang dibangun AI ini ternyata sangat menarik dan mempercepat proses pemahaman terhadap kecerdasan manusia [**13**]. Stuart Russel dan Peter Norvig mengelompokan definisi AI ke dalam empat kategori, yaitu :

2.3.1 *Thinking Humanly*

Pendekatan ini dilakukan dengan dua cara, yaitu :

1. Melalui introspeksi : mencoba menangkap pemikiran kita sendiri pada saat berfikir. Tetapi seorang psikolog barat mengatakan *"how do you know that you understand?"* bagaimana anda tahu bagaimana anda mengerti? Karena pada saat anda menyadari pemikiran anda, ternyata pemikiran tersebut sudah lewat dan digantikan kesadaran anda. Sehingga definisi ini terkesan mengada-ada dan tidak mungkin dilakukan.
2. Melalui eksperimen psikologi

2.3.2 *Acting Humanly*

Pada tahun 1950, Alan Turing merancang suatu ujian bagi komputer berintelejensia untuk menguji apakah komputer tersebut mampu mengelabuhi seorang manusia yang menginterogasinya melalui *teletype* (komunikasi berbasis teks jarak jauh).

2.3.3 *Thinking Rationally*

Terdapat dua masalah dalam pendekatan ini, yaitu :

1. Tidak mudah untuk membuat pengetahuan informasi *informal* dan menyatakan pengetahuan tersebut ke dalam *formal term* yang diperlukan oleh notasi logika, khususnya ketika pengetahuan tersebut memiliki kepastian kurang dari 100%.
2. Terdapat perbedaan besar antara dapat memecahkan masalah “dalam prinsip” dan memecahkannya “dalam dunia nyata”.

2.3.4 *Acting Rationally*

Membuat inferensi yang logis merupakan bagian dari suatu *rational agent*. Hal ini disebabkan satu-satunya cara untuk melakukan aksi secara rasional adalah dengan menalar secara logis. Dengan menalar secara logis, maka bisa didapatkan kesimpulan bahwa aksi yang diberikan akan mencapai tujuan atau tidak. Jika mencapai tujuan, maka *agent* bisa melakukan aksi berdasarkan kesimpulan tersebut.

Thinking humanly dan *acting humanly* adalah dua definisi dalam arti yang sangat luas. Sampai saat ini, pemikiran manusia yang diluar rasio, yakni refleks dan intuitif (berhubungan dengan perasaan), belum dapat ditirukan sepenuhnya oleh komputer. Dengan demikian, kedua definisi ini dirasa kurang tepat untuk saat ini. Jika kita menggunakan definisi ini, maka banyak produk komputasi cerdas saat ini yang tidak layak disebut sebagai produk AI.

1. Definisi *thinking rationally* terasa lebih sempit daripada *acting rationally*. Oleh karena itu, definisi AI yang paling tepat untuk saat ini adalah *acting rationally* dengan pendekatan *normal agent*. Hal ini berdasarkan

pemikiran bahwa komputer bisa melakukan penalaran secara logis dan juga bisa melakukan aksi secara rasional berdasarkan hasil penalaran tersebut [**13**].

2.4 *Machine Learning*

Komputer dibangun untuk dapat menjalankan tugas atau menyelesaikan masalah berdasarkan instruksi dari pengguna. Dalam proses menjalankan tugas, komputer menggunakan algoritma. Algoritma adalah sekumpulan instruksi yang tersusun secara urut. Fungsi algoritma adalah melakukan transformasi dari data *input* menjadi data *output*. *Machine learning* mempunyai pendekatan agar sebuah komputer tidak lagi hanya bisa menjalankan algoritma, akan tetapi dapat membuat algoritma sendiri. Proses pembuatan algoritma dapat dilakukan dengan proses pembelajaran (*learning*) dengan mempertimbangkan data yang ada atau dengan kata lain komputer diupayakan untuk dapat mengekstrak algoritma secara otomatis berdasarkan data yang ada untuk menyelesaikan tugasnya [**14**].

2.4.1 *Supervised Learning*

Supervised learning adalah teknik pembelajaran mesin dengan membuat suatu fungsi dari data latihan. Data latihan terdiri dari pasangan nilai *input* dan *output* yang diharapkan dari *input* yang bersangkutan. Tugas dari *supervised learning* adalah untuk memprediksi nilai fungsi untuk nilai semua input yang ada [**14**].

2.4.2 *Unsupervised Learning*

Teknik ini menggunakan prosedur yang berusaha untuk mencari partisi dari sebuah pola. *Unsupervised learning* mempelajari bagaimana sebuah sistem dapat belajar untuk merepresentasikan pola input dalam cara yang menggambarkan struktur statistikal dari keseluruhan pola input. Berbeda dari *supervised learning*, *unsupervised learning* tidak memiliki target *output* yang eksplisit atau tidak ada pengklasifikasian *input* [**14**].

2.5 *Pattern Recognition*

Pattern recognition merupakan salah satu bidang dalam *computer sains*, yang memetakan suatu data ke dalam konsep tertentu. Konsep tertentu ini disebut *class* atau *category*. Aplikasi *pattern recognition* sangat luas, diantaranya mengenali suara dalam sistem keamanan. Membaca huruf dalam OCR, mengklasifikasikan penyakit, dan lain sebagainya. Berbagai metode yang ada pada *pattern recognition*, seperti *linear discriminant analysis, hidden markov model* hingga metode kecerdasan buatan seperti jaringan syaraf tiruan. Salah satu metode yang banyak mendapat perhatian sebagai *state of the art* dalam *pattern recognition* adalah *support vector machine* [**15**].

2.6 *Support Vector Machine*

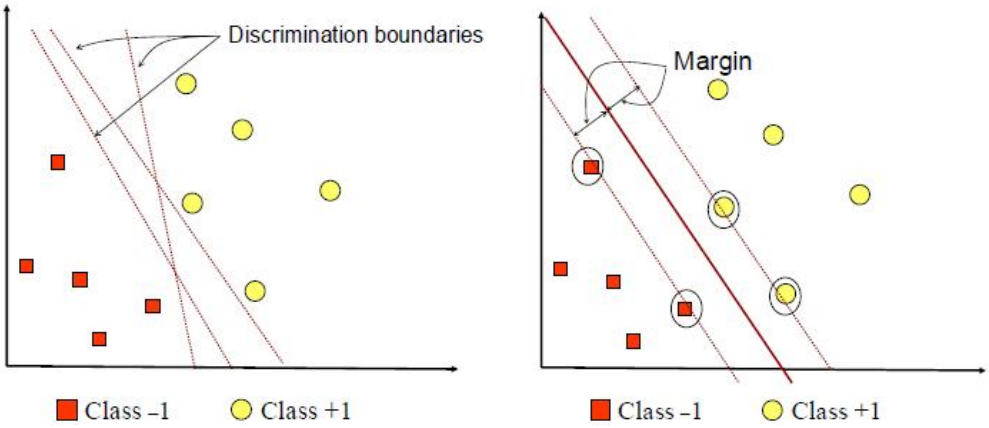
Support vector machine (SVM) muncul pertama kali pada tahun 1992 oleh Vladimir Vapnik bersama rekannya Bernhard Boser dan Isabelle Guyon. *Support vector machine* (SVM) merupakan metode klasifikasi jenis terpandu (*supervised*) karena ketika pelatihan, diperlukan target pembelajaran tertentu. SVM dapat digunakan untuk klasifikasi yang dapat diterapkan pada deteksi tulisan tangan, pengenalan obyek, identifikasi suara dan lain-lain [**4**][**16**].

SVM adalah metode *machine learning* yang bekerja atas prinsip *Structural Risk Minimization* (SRM) dengan tujuan menemukan *hyperplane* terbaik yang memisahkan dua buah *class* pada *input space.*

Berbeda dengan strategi *neural network* yang berusaha mencari *hyperplane* pemisah antar *class*, SVM berusaha menemukan *hyperplane* yang terbaik pada *input space.* Konsep SVM dapat dijelaskan secara sederhana sebagai usaha mencari *hyperplane* terbaik berfungsi sebagai pemisah dua buah *class* pada *input space* [**16**].

2.6.1 Kasus Data yang Terpisah Secara *Linear*

Kasus data yang terpisah secara *linear* adalah kasus yang termudah, seperti yang terlihat pada Gambar 2.5 di bawah ini.



Gambar 2.5 SVM Berusaha Menemukan *Hyperplane* Pemisah

Gambar 2.5 memperlihatkan beberapa *pattern* yang merupakan anggota dari dua buah *class* : +1 dan -1 yang mempunyai tupel pelatihan 2-D. *Pattern* yang tergabung pada *class* -1 disimbolkan dengan kotak berwarna merah sementara *pattern* pada *class* +1 disimbolkan dengan lingkaran berwarna kuning. Masalah klasifikasi dapat diterjemahkan dengan usaha menemukan *hyperplane* yang memisahkan antara kedua kelompok tersebut [**16**].

Hyperplane pemisah terbaik antara kedua *class* dapat ditemukan dengan mengukur *margin hyperplane* dengan *pattern* terdekat masing-masing *class*. *Pattern* yang paling dekat ini disebut dengan *support vector*. Garis solid pada Gambar 2.5 sebelah kanan menunjukan *hyperplane* terbaik, yaitu yang terletak pada tengah-tengah kedua *class*, sedangkan titik merah dan kuning yang berada dalam lingkaran hitam adalah *support vector* [**16**].

Data yang tersedia dinotasikan sebagai $x \in R^d$ sedangkan label masing-masing dinotasikan $y_i \in \{-1,+1\}$ untuk $i = 1,2,\ldots n$ yang mana n adalah banyaknya data. Diasumsikan kedua *class* dapat terpisah secara sempurna oleh *hyperplane* berdimensi d, yang didefinisikan pada persamaan 2-1.

$$w.x + b = 0 \qquad (2\text{-}1)$$

Pattern x_i yang terdapat pada *class* -1 dapat dirumuskan sebagai *pattern* yang memenuhi persamaan 2-2.

$$w.x + b \leq -1 \quad (2\text{-}2)$$

Pattern x_i yang termasuk *class* +1 dapat dirumuskan dengan persamaan 2-3.

$$w.x + b \geq 1 \quad (2\text{-}3)$$

Keterangan :

w = *vector* bobot

x = nilai masukan atribut

b = *bias*

Margin terbesar dapat ditemukan dengan memaksimalkan nilai jarak antara jarak dan titik terdekatnya, yaitu $1/||w||$. Hal ini dapat dirumuskan sebagai *quadratic programming* (QP) *problem*, yaitu mencari titik minimal yang dinyatakan dalam persamaan 2-4 dan memperhatikan kondisi yang harus dipenuhi pada persamaan 2-5.

$$min_w = \tau(w) = 1/2||w||^2 \quad (2\text{-}4)$$

$$y_i(x_i.w + b) - 1 \geq 0 \quad (2\text{-}5)$$

Problem ini dapat dipecahkan dengan berbagai teknik komputasi, diantaranya *lagrange multiplier* yang dinyatakan pada persamaan 2-6.

$$L(w, b, a) = \frac{1}{2||w||^2} - \sum_{i=1}^{l} a_i y_i(x_i.w + b) - 1 \text{ dengan } i = 1,2 \ldots l \quad (2\text{-}6)$$

Dimana a_i adalah *lagrange multiplier*, yang bernilai 0 atau positif $a_i \geq 0$. Nilai optimal dari persamaan 2-6 dapat dihitung dengan meminimalkan L terhadap w dan b, dan memaksimalkan L terhadap a_i. Dengan memperhatikan sifat bahwa pada titik optimal *gradient* L=0 persamaan 2-5 dapat dimodifikasi sebagai maksimasi *problem* yang hanya mengandung a_i, sebagaimana terlihat pada persamaan 2-7 dan 2-8.

$$\sum_{i=1}^{l} a_i - \frac{1}{2}\sum_{i,j=1}^{l} a_i a_j y_i y_j x_i x_j \quad (2\text{-}7)$$

$$\sum_{i=1}^{l} a_i y_i = 0 \text{ untuk } a_i \geq 0 (i = 1,2,..l) \quad (2\text{-}8)$$

Dengan demikian, maka akan diperoleh a_i yang kebanyakan bernilai positif yang disebut sebagai *support* vector [**16**].

2.6.2 Kasus Data yang Tidak Terpisah Secara *Linear*

Kasus data yang tidak terpisah secara *linear* diasumsikan bahwa *class* pada *input space* tidak dapat terpisah secara sempurna. Hal ini menyebabkan *constraint* pada persamaan 2-5 tidak dapat terpenuhi, sehingga optimalisasi tidak dapat dilakukan, untuk mengatasi masalah ini SVM dirumuskan ulang dengan memperkenalkan teknik *softmargin*. Dalam *softmargin* persamaan 2-5 dimodifikasi dengan menggunakan *slack* variabel sehingga terlihat pada persamaan 2-9 [**16**].

$$y_i(x_i.w + b) \geq 1 - \varepsilon_i \quad (2\text{-}9)$$

Dengan demikian persamaan 2-4 diubah menjadi persamaan 2-10.

$$min_w = \tau(w) = \frac{1}{2\|w\|^2} + C\sum_{i=1}^{l} \varepsilon_i \quad (2\text{-}10)$$

Fitur *C* digunakan untuk mengontrol *tradeoff* antara *margin* dan kesalahan klasifikasi ε.

2.6.3 *Kernel Trick* dan *Non-Linear Classification* Pada SVM

Pada umumnya masalah yang terjadi dalam dunia nyata jarang yang bersifat *linear separable*. Kebanyakan bersifat *non-linear*, SVM dimodifikasi dengan memasukan fungsi *kernel* [**4**].

Dalam non *linear* SVM, pertama-tama data *x* dipetakan oleh fungsi *Φ (x)* ke ruang vektor yang berdimensi lebih tinggi. Pada ruang vektor yang baru ini, *hyperplane* yang memisahkan kedua *class* tersebut dapat dikonstruksikan. Hal ini sejalan dengan teori Cover yang menyatakan "*Jika suatu transformasi bersifat non linear dan dimensi dari feature space cukup tinggi, maka data pada input*

space dapat dipetakan ke feature space yang baru, dimana pattern-pattern tersebut pada probabilitas tinggi dapat dipisahkan secara linear" .

Pemetaan ini dilakukan dengan menjaga topologi data, dalam artian dua data yang berjarak dekat pada *input space* akan berjarak dekat juga pada *feature space*, sebaliknya dua data yang berjarak jauh pada *input space* akan juga berjarak jauh pada *feature space*.

Selanjutnya proses pembelajaran pada SVM dalam menemukan titik-titik *support vector*, hanya bergantung pada *dot product* dari data yang sudah ditransformasikan pada ruang baru yang berdimensi lebih tinggi, yaitu $\Phi(x_i).\Phi(x_j)$ [**16**].

Karena umumnya transformasi Φ ini tidak diketahui, dan sangat sulit untuk difahami secara mudah, maka perhitungan *dot product* tersebut sesuai teori Mercer dapat digantikan dengan fungsi *kernel* yang terlihat pada persamaan 2-11.

$$K(x_i, x_j) = \Phi(x_i).\Phi(x_j) \quad (2\text{-}11)$$

Beberapa *kernel* yang terdapat pada svm meliputi :

1. Polinomial Derajat h

   Kernel trick polinomial cocok digunakan untuk menyelesaikan masalah klasifikasi, dimana dataset pelatihan sudah normal. *Kernel trick* ini dinyatakan dalam persamaan 2-12.

   $$K(x_i, x_j) = (x_i.x_j + 1)^h \quad (2\text{-}12)$$

2. *Radial Basis Function*

   Kernel trick radial basis function merupakan *kernel* yang paling banyak digunakan untuk menyelesaikan masalah klasifikasi untuk dataset yang tidak terpisah secara *linear*, dikarenakan akurasi pelatihan dan akurasi prediksi yang sangat baik pada *kernel* ini, dimana *kernel radial basis function* dinyatakan dalam persamaan 2-13.

   $$K(x_i, x_j) = e^{-||x_i - x_i||^2 / 2\sigma^2} \quad (2\text{-}13)$$

3. *Sigmoid*

   Kernel sigmoid merupakan *kernel trick* svm yang merupakan pengembangan dari jaringan saraf tiruan, dimana *kernel* ini dinyatakan dengan persamaan 2-14.

$$K(x_i, x_j) = \tanh(kx_i . x_j - \delta) \quad (2\text{-}14)$$

Kernel trick memberikan beberapa kemudahan, karena dalam proses pembelajaran SVM, untuk menentukan *support vector*, pengguna hanya cukup mengetahui fungsi *kernel trick* yang dipakai, tanpa perlu mengetahui wujud dari fungsi *non-linear* [**4**][**16**].

Dari keseluruhan *kernel trick* tersebut, *kernel trick radial basis function* merupakan *kernel trick* yang memberikan hasil terbaik pada proses klasifikasi khususnya untuk data yang tidak bisa dipisahkan secara *linear* [**16**].

Selain masalah data yang tidak dipisahkan secara *linear* ada masalah lain yang sering muncul dalam penerapan metode klasifikasi *machine learning* seperti *support vector machine* adalah masalah dimensionalitas dataset atau sering disebut sebagai kutukan dimensionalitas (*curse of dimensionality*), jika dimensi meningkat, data akan meningkat secara halus dalam daerah yang ditempati. Untuk itu diperlukan pengurangan dimensi.

Manfaat dari pengurangan dimensi :

1. Mencegah terjadinya efek dari dimensionalitas.
2. Mengurangi jumlah waktu dan memori yang dibutuhkan oleh *machine learning.*
3. Membuat data lebih mudah divisualisasikan.
4. Membantu untuk mengurangi fitur-fitur yang tidak relevan atau mengurangi gangguan/derau.

Teknik pengurangan dimensionalitas data diantaranya adalah *principal component analysis* (PCA), standar deviasi, *zero-mean*, *min-max normalization*, dan lain-lain [**17**][**18**].

2.7 Standar Deviasi

Standar deviasi disebut juga simpangan baku merupakan metode untuk mencari variasi suatu data dan merupakan metode yang digunakan untuk mengurangi dimensionalitas dari suatu dataset dan mempunyai satuan ukuran yang sama dengan data asal. Singkatnya, standar deviasi mengukur-bagaimana nilai-nlai data tersebar, bisa juga didefinisikan sebagai rata-rata jarak penyimpangan titik-titik. Standar deviasi merupakan hasil akar dari pengurangan dataset dengan nilai rata-rata dari dataset dibagi dengan jumlah dataset. Standar deviasi ini ditulis dengan persamaan 2-15.

$$s = \sum \sqrt{\frac{|x_i - \bar{x}|^2}{n-1}} \qquad (2\text{-}15)$$

Keterangan :

$\bar{x}$ = rataan hitung

x_i = *input* data

n = jumlah data

s = standar deviasi

2.8 Klasifikasi Sinyal Berdasarkan Eksitasi

Berdasarkan eksitasi yang dihasilkan pada proses produksi suara, sinyal suara dapat dibagi menjadi tiga bagian, yaitu *silence*, *unvoiced*, dan *voiced.*

1. Sinyal *silence*

   Sinyal *silence* merupakan sinyal pada saat tidak terjadi proses produksi suara ucapan dan sinyal yang diterima oleh pendengar dianggap sebagai bising latar belakang.

2. Sinyal *unvoiced*

   Sinyal *unvoiced* terjadi pada saat pita suara tidak bergetar, dimana sinyal eksitasi berupa sinyal *random*.

3. Sinyal *voiced*

   Sinyal *voiced* terjadi jika pita suara bergetar, yaitu pada saat sinyal eksitasi berupa sinyal pulsa quasi-periodik. Selama terjadinya sinyal *voiced* ini, pita suara bergetar pada frekuensi fundamental [**7**].

2.9 Sinyal Digital dan Sinyal Analog

Sinyal itu ialah besaran yang berubah dalam waktu dan atau dalam ruang, dan membawa suatu informasi. Pada umumnya variabel independen untuk sinyal adalah waktu. Jika variabel independennya kontinu, maka sinyal tersebut disebut sebagai sinyal waktu kontinu (*continuous-time signal*). Jika variabel independennya diskrit, maka sinyal tersebut disebut sebagai sinyal waktu diskrit (*discrete-time signal*). Sinyal waktu kontinu didefinisikan setiap waktu dalam sebuah interval yang biasanya tidak terbatas, sedangkan sinyal waktu diskrit didefinisikan pada waktu diskrit, dan biasanya berupa urutan angka.

Sinyal waktu kontinu dengan amplitudo kontinu biasanya disebut sebagai sinyal analog. Contoh sinyal analog adalah sinyal suara. Sinyal waktu diskrit dengan amplitudo bernilai diskrit yang direpresentasikan oleh digit angka yang terbatas (*finite*), biasanya disebut sebagai sinyal digital [**8**].

2.9.1 Sinyal Digital

Sinyal digital merupakan hasil teknologi yang dapat mengubah sinyal menjadi kombinasi urutan bilangan 0 dan 1 (juga dengan biner), sehingga tidak mudah terpengaruh oleh derau, proses informasinya pun mudah, cepat dan akurat, tetapi transmisi dengan sinyal digital hanya mencapai jarak jangkau pengiriman data yang relatif dekat. Biasanya sinyal ini juga dikenal dengan sinyal diskrit. Sinyal yang mempunyai dua keadaan ini biasa disebut dengan bit. Bit merupakan istilah khas pada sinyal digital. Sebuah bit dapat berupa nol (0) atau satu (1). Kemungkinan nilai untuk sebuah bit adalah 2 buah (2^1). Kemungkinan nilai untuk 2 bit adalah sebanyak 4 (2^2), berupa 00, 01, 10, dan 11. Secara umum, jumlah kemungkinan nilai yang terbentuk oleh kombinasi n bit adalah sebesar 2^n buah.

Sinyal digital merupakan bentuk *sampling* dari sinyal analog digital pada dasarnya di *code*-kan dalam bentuk *biner* (atau *hexa*) besarnya nilai suatu sistem

digital dibatasi oleh lebarnya/jumlah bit *(bandwidth).* Jumlah bit juga sangat mempengaruhi nilai akurasi dan kualitas sinyal digital [**8**].

2.9.2 Sinyal Analog

Sinyal analog atau sinyal waktu kontinyu adalah sinyal yang memiliki nilai real pada setiap waktu. Sinyal kontinyu merupakan suatu sinyal yang berbentuk gelombang sinusoidal dan merupakan variabel yang berdiri sendiri. Pada sinyal kontinyu, variabel *indipendent* (yang berdiri sendiri) terjadi terus-menerus dan kemudian sinyal dinyatakan sebagai sebuah kesatuan nilai dari variabel *independent*. Dengan menggunakan sinyal analog, maka jangkauan transmisi data dapat mencapai jarak yang jauh, tetapi sinyal ini mudah terpengaruh oleh *noise*.

Gelombang pada sinyal analog yang umumnya berbentuk gelombang sinus memiliki tiga variable dasar, yaitu amplitudo, frekuensi, dan *phase* [**8**].

1. Amplitudo merupakan ukuran tinggi rendahnya tegangan dari sinyal analog.
2. Frekuensi adalah jumlah gelombang sinyal analog dalam satuan detik.
   Phase adalah besar sudut dari sinyal analog pada saat tertentu.

2.10 Jenis *File Audio Digital*

Setiap bentuk *file* audio memiliki kelebihan dan kekurangan masing-masing. Format *file* audio tersebut dapat dirubah sesuai dengan kebutuhan. Format *file* audio bermacam-macam, diantaranya :

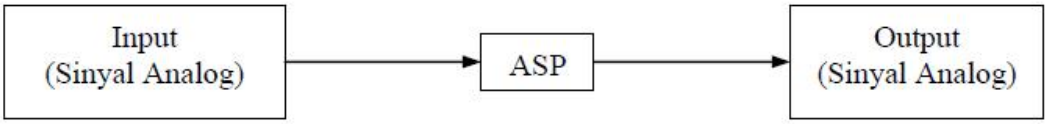
1. WAV *(Wave),* format *file* ini merupakan dasar dari format audio *file* yang memiliki kualitas suara terbaik, hanya saja *file* ini membutuhkan tempat penyimpanan yang besar. Pemilihan format ini sangat tepat apabila membutuhkan kualitas audio yang baik dan memiliki tempat penyimpanan yang besar. Format *file* ini mendukung untuk mono atau stereo.
2. Apple AIFF (.AIF, .SND), format ini adalah format audio standar milik Apple Computer. Seperti WAV milik Windows, AIFF mendukung untuk fasilitas mono atau stereo, 16-bit atau 8-bit.

3. Dialogic ADPCM (.VOX), format Dialogic ADPCM ini biasanya ditemui pada aplikasi telepon. Format ini hanya dapat menyimpan audio mono 16-bit, dan seperti format ADPCM lainnya *file* ini dapat dikompres hingga 4-bit.
4. DiamondWare Digitized (.DWD), ini adalah format audio yang digunakan oleh perangkat DiamondWare's Sound, biasanya format ini digunakan oleh para programmer untuk menghasilkan audio interaktif yang diaplikasikan pada game dan multimedia. Format ini juga medukung baik mono maupun stereo.
5. MPEG Audio Player 3 (.MP3), ini merupakan format audio *file* yang banyak diminati oleh para pengguna komputer, karena disamping kualitas yang dihasilkan baik *file* ini juga tidak memerlukan tempat penimpanan yang besar.
6. Real Media (.RM), format audio ini biasanya dapat ditemukan pada jaringan internet.
7. Sound Blaster (.VOC), ini adalah format audio *file* dari Sound Blaster dan format *file* suara dari Sound Blaster Pro. Format ini hanya mendukung 8-bit audio, mono hingga 44.1 KHz, dan stereo hingga 22 KHz.
8. *Advance Audio Coding (AAC).* Sepuluh tahun sejak ditemukannya MP3, sering pula disebut MP4. Apel merupakan vendor yang paling getol menggunakan *file* suara berlisensi ini. Apel juga merupakan pengembang dari *file* AAC ini yang bisa dijalankan di iTunes, QuickTime 6, iPod, dan seterusnya.
9. Musik Digital (MIDI) merupakan standar untuk menghubungkan komputer dengan instrumen musik elektronik dan pemrosesan efek khusus. Format suara instrumen ini di perkenalkan pada tahun 1983 oleh perusahaan musik elektrik seperti Roland, Yamaha, dan Korg. Format MIDI bersifat sangat kompak dengan ukurannya yang kecil, suara yang di hasilkan oleh MIDI dengan dukungan sound card yang memiliki *synthesizer* (penghasil suara elektrik) sangatlah mirip dengan organ elektrik yang bisa memainkan berbagai alatmusik atau peranti elektronik

lainnya, tetapi tidak cocok untuk hasil konversi dari suara analog karena tidak terlaluakurat. *File* dengan format ini berukuran kecil dan sering digunakan dalam ponsel sebagai *ringtone* [**8**].

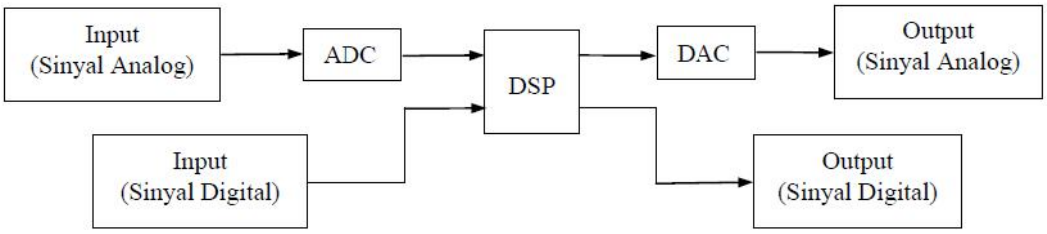
2.11 Elemen Dasar Sistem Pemrosesan Sinyal

Dalam proses pengolahan sinyal analog, sinyal input masuk ke Analog Signal Processing (ASP), untuk kemudian diberi berbagai perlakuan (misalnya pemfilteran, penguatan, dan sebagainya) dan outputnya berupa sinyal analog seperti yang terlihat pada Gambar 2.6.



Gambar 2.6 Sistem Pengolahan Sinyal Analog

Proses pengolahan sinyal secara digital memiliki alur sedikit berbeda. Komponen utama sistem ini berupa sebuah *processor* digital yang mampu bekerja apabila inputnya berupa sinyal digital. Untuk sebuah input berupa sinyal analog perlu proses awal yang bernama digitalisasi melalui perangkat yang bernama *analog-to-digital conversion* (ADC), dimana sinyal analog harus melalui proses *sampling*, kuantisasi dan pengkodean (*coding*). Demikian juga output dari processor digital harus melalui perangkat *digital-to-analog conversion* (DAC) agar outputnya kembali menjadi bentuk analog. Ini bisa kita amati pada perangkat seperti PC, *digital sound system*, dan sebagainya. Secara sederhana bentuk blok diagramnya adalah terlihat seperti pada Gambar 2.7.



Gambar 2.7 Sistem Pengolahan Sinyal Digital

2.11.1 Pengubahan Sinyal Analog Menjadi Sinyal Digital

Sebagian besar sinyal, seperti sinyal suara merupakan sinyal analog, untuk memproses sinyal analog dengan alat digital, tidak dapat diproses begitu saja atau dengan kata lain sinyal analog harus terlebih dahulu dirubah menjadi sinyal digital. Proses ini dinamakan konversi analog ke digital (A/D) dengan alat yang dinamakan *analog to digital converter* (ADC).

Terdapat tiga tahap dalam pengubahan sinyal analog menjadi sinyal digital, seperti terlihat pada Gambar 2.8.



Gambar 2.8 Tahapan ADC (*Analog to Digital Converter*)

Penjelasan tahapan ADC (*analog to digital converter*) pada Gambar 2.8 adalah sebagai berikut :

1. *Sampling*

   Sampling adalah konversi sinyal kontinu dalam domain waktu menjadi sinyal diskrit melalui proses sampling sinyal pada selang waktu tertentu.

2. Kuantisasi

   Kuantisasi merupakan proses pemetaan dari nilai sinyal kontinyu dari hasil sampling menjadi nilai-nilai yang diskrit sehingga didapatkan sinyal nilai diskrit.

3. *Coding*

   Tiap nilai diskrit yang telah didapat, direpresentasikan dengan angka binary n-bit.

2.12 Pemrosesan Sinyal Suara

Pemrosesan sinyal suara bermaksud mengubah karakteristik sinyal suara atau mengambil beberapa informasi yang diinginkan dari sinyal suara.

2.12.1 *Pre-emphasis*

Pre-emphasis dilakukan untuk menghilangkan informasi yang tidak relevan dan kebisingan dengan menggunakan perhitungan *low pass filter*. *Pre-*

emphasis merujuk pada proses memaksimalkan kualitas sinyal dengan meminimalkan efek *noise* seperti distorsi selama perekaman dan transmisi data, serta memperhalus bentuk *spectral* frekuensi [**9**].

Pre-emphasis didasari oleh hubungan input dan output dalam domain waktu yang dinyatakan dalam persamaan 2-16.

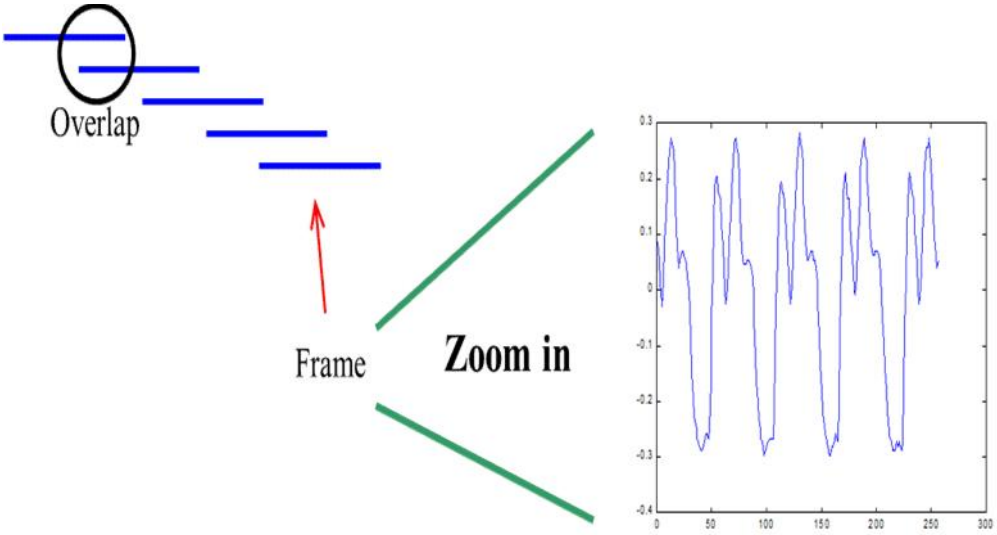
$$y(n) = x(n) - ay(n-1) \quad (2\text{-}16)$$

Keterangan :

y(n)	=	sinyal hasil *pre-emphasis*
x(n)	=	sinyal masukan
y(n-1)	=	hasil *pre-emphasis* segmen suara sebelumnya
a	=	konstanta filter *pre-emphasis* yang bernilai 0,9

2.12.2 *Frame Blocking*

Sinyal suara dibagi-bagi ke dalam *frame* dengan panjang segmen per-*frame* adalah Z dan dipisahkan sejauh *overlap* Y dengan Y<Z [**10**]. Gambar di bawah ini merupakan *frame blocking.*



Gambar 2.9 *Frame Blocking*

2.12.3 *Hamming Window*

Windowing diperlukan untuk mengurangi efek diskontinuitas dari potongan sinyal. Dimana jenis *windowing* ada beberapa macam, yaitu *Hamming*, *Hanning*, *Bartlet*, *Rectanguler*, dan *Blackman*. Metode *windowing* yang dipakai untuk pemrosesan sinyal suara adalah *hamming window*, dimana *hamming window* dinyatakan dalam persamaan 2-17 dan perkalian sinyal suara dengan *hamming window* dinyatakan dalam persamaan 2-18 [**11**].

$$w(n) = 0.54 - 0.46\cos\left(2\pi \frac{n}{N}\right) \quad (2\text{-}17)$$

$$y(n) = x(n) * w(n) \quad (2\text{-}18)$$

Keterangan :

y(n)	=	hasil perkalian segmen sinyal suara dengan *hamming window*
x(n)	=	*input* segmen suara
w(n)	=	nilai *hamming window*
π	=	22/7 atau 3.14
n	=	nomor segmen suara
N	=	panjang segmen suara – 1

2.12.4 Metode *Autocorrelation*

Autocorrelation adalah korelasi silang terhadap sinyal dengan dirinya sendiri. Nilai autokorelasi suatu sinyal wicara akan menunjukkan bagaimana gelombang suara itu membentuk suatu korelasi kepada dirinya sendiri. Bentuk-bentuk yang sama pada setiap *delay* waktu tertentu menunjukkan perulangan bentuk pola sinyal suara. Dengan demikian akan bisa dilakukan estimasi nilai frekuensi fundamentalnya [**2**]. *Autocorrelation* dinyatakan dengan persamaan 2-19 dan 2-20, dengan *vector output* pada persamaan 2-21, sementara untuk menghitung frekuensi *pitch* dinyatakan dengan persamaan 2-22.

$$Rxx(m) = \sum_{n=0}^{N-m-1} x(n) \cdot x(n+m), \ m \geq 0 \quad (2\text{-}19)$$

$$Rxx(m) = Rxx(-m), m < 0 \quad (2\text{-}20)$$

$$c(m) = Rxx(m) / Rxx(0) \quad (2\text{-}21)$$

$$f_0 = 1 / T_0 \quad (2\text{-}22)$$

Keterangan :

Rxx(m)	=	nilai autokorelasi
c(m)	=	*vektor output*
m	=	Maxlag
x(n)	=	input sinyal suara
N	=	panjang sinyal suara
f_0	=	frekuensi fundamental
T_0	=	periode frekuensi fundamental

2.12.5 Metode *Linear Predictive Coding*

Linear predictive coding merupakan salah satu metode pemodelan suara yang didasarkan pada teori bahwa sebuah sinyal suara manusia pada waktu n, s(n), dapat diperkirakan sebagai kombinasi linear dari p sinyal suara manusia sebelumnya. Tujuan dari metode lpc adalah untuk memisahkan efek *formant* dengan *pitch* atau frekuensi dasar manusia. Hal ini direpresentasikan dengan menggunakan persamaan 2-23 [**10**].

$$s_n = \sum_{i=1}^{p} a_i . s_{(n-i)} \quad (2\text{-}23)$$

Langkah pertama dalam mencari koefisien lpc dengan menggunakan metode *linear predictive coding* adalah melakukan autokorelasi sinyal suara dengan *maxlag* adalah *p*. Setelah nilai autokorelasi didapat, langkah selanjutnya adalah melakukan perhitungan *levinson durbin* pada nilai autokorelasi yang didapat, seperti terlihat pada persamaan 2-24 untuk mendapatkan koefisien lpc A(z) [**9**].

Levinson Durbin :

$$\begin{bmatrix} R[0] & R[1] & R[2] & R[p] \\ R[1] & R[0] & R[1] & R[p-1] \\ R[2] & R[1] & R[0] & R[p-2] \\ R[p] & R[p-1] & R[p-2] & R[0] \end{bmatrix} \begin{bmatrix} a1 \\ a2 \\ a3 \\ ap \end{bmatrix} = \begin{bmatrix} R(1) \\ R(2) \\ R(3) \\ R(p) \end{bmatrix} \quad (2\text{-}24)$$

Dengan solusi pemecahan terlihat pada persamaan 2-25 sampai dengan persamaan 2-31.

$$E^{(0)} = R(0) \quad (2\text{-}25)$$

$$k_{(i)} = R(i) - \sum_{j=1}^{i-1} a_j^{(i-1)} R(i-j) / E^{(i-1)} \quad (2\text{-}26)$$

$$a_i^{(i)} = k_{(i)} \quad (2\text{-}27)$$

$$a_j^{(i)} = a_j^{(i-1)} - k_{(i)} a_{i-j} \quad (2\text{-}28)$$

$$E^{(i)} = (1 - k_i^i) E^{(i-1)} \quad (2\text{-}29)$$

$$a_j = a_j^p \quad (2\text{-}30)$$

$$A(z) = 1 - \sum_{k=1}^{p} a_k z^{-k} \quad (2\text{-}31)$$

Keterangan :

$R(i)$ = nilai autokorelasi

$E^{(i)}$ = *error*

$k_{(i)}$ = koefisien pantulan

$a_{(j)}$ = koefisien lpc

$A(z)$ = persamaan koefisien lpc

Setelah persamaan koefisien didapat, maka selanjutnya adalah mencari nilai r_0 atau akar-akar persamaan koefisien A(z) dan kemudian dirubah ke dalam $\bar{}_0$ sering juga disebut sebagai perubahan sudut untuk kemudian dihitung nilai *formant* dan *bandwidth* dengan persamaan 2-32 dan 2-33 [**12**].

$$F = \frac{fs}{2®} - _0 \quad (2\text{-}32)$$

$$B = -\frac{fs}{®} \ln r_0 \quad (2\text{-}33)$$

Keterangan :

F = frekuensi *formant*

fs = frekuensi pencuplikan

® = konstanta 22/7 atau 3.14

$_{0}$ = perubahan sudut

B = *bandwith*

r_0 = akar-akar persamaan $A(z)$