SPATIAL AND TEMPERATURE DEPENDENCE OF MAGNETIC DOMAIN MEMORY INDUCED BY EXCHANGE-COUPLING

by

Joseph A. Nelson

A senior thesis submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Bachelor of Science

Department of Physics and Astronomy

Brigham Young University

April 2010

BRIGHAM YOUNG UNIVERSITY

DEPARTMENT APPROVAL

of a senior thesis submitted by

Joseph A. Nelson

This thesis has been reviewed by the research advisor, research coordinator, and department chair and has been found to be satisfactory.

Date	Karine M. Chesnel, Advisor
Date	Eric Hintz, Research Coordinator
Date	Ross Spencer, Chair

ABSTRACT

SPATIAL AND TEMPERATURE DEPENDENCE OF MAGNETIC DOMAIN MEMORY INDUCED BY EXCHANGE-COUPLING

Joseph A. Nelson

Department of Physics and Astronomy

Bachelor of Science

Exchange-bias [CoPd]/IrMn thin films with perpendicular magnetization have shown evidence of magnetic domain memory [1]. Using cross-correlation metrology on x-ray resonant magnetic scattering data, together with magnetometry measurements performed here at BYU, we have quantified this memory, and have determined its dependence on field cycling (field magnitude across the magnetic hysteresis loop). We have also studied how this domain memory changes with spatial scales and with temperature. We find the domain memory to be over 95% at the spatial scale corresponding to the average domain periodicity. This memory remains strong even after repeated field cycling, and decreases very little with increasing temperature, remaining over 90% at temperatures as high as 220K. Finally, we find evidence for a spatial superstructure in the memory, and suggest that this behavior results from an interaction between disorder-induced memory and exchange-bias-induced memory.

ACKNOWLEDGMENTS

I would like to thank Dr. Karine Chesnel for her help, patience, and support. It has been a wonderful experience working with her.

I would also like to acknowledge both BYU's Office of Research and Creative Activities (ORCA) and the Department of Physics for the financial support that made this research possible.

Most of all, I thank my patient and supportive wife Leah, who has quietly shouldered most of the burdens of caring for our family while I studied and worked. She has been both a soft place to fall and a rock of support. Thank you.

Contents

List of Figures

Chapter 1

Introduction to Magnetic Domain Memory

1.1 Motivation

The areal storage density of magnetic recording media has been increasing exponentially every year for over fifty years. The annual increase rate has never been below 25%, and has been over 60% per year since 1991 [2]. Increased memory demands for internet, intelligence, bioinformatics, and even mobile devices require continuously higher memory storage density. These demands have led to the investigation of the properties of thin magnetic films. In such films, when the thickness of magnetic layers is finely adjusted, the magnetization tends to prefer perpendicular orientation rather than in-plane [3]. Such systems exhibit ferromagnetic domains, i.e. regions where magnetic moments are aligned as explained in the next section. The size of these domains is a crucial parameter for the potential storage density of the material, and make thin films attractive objects of study.

Apart from storage density requirements, the greatest challenge facing the data storage industry is the sensitivity of magnetic media to degradation by external fields. Because of this sensitivity, long-term archival data storage, which is required by law for many industries, has become very expensive to upkeep. To protect against possible degradation, most archives must be removed, read, and re-recorded every few years. This process is very labor-intensive, time consuming, and expensive. This has led to the search for magnetic systems which have intrinsic magnetic memory, or ability to reproduce a former morphology after saturation by an external field.

1.2 Ferromagnetism, Hysteresis, and Memory

The atoms of ferromagnetic materials, such as Cobalt, Iron, or Nickel, have intrinsic magnetic spins. The magnetic coupling between spins is so strong that they tend to align, creating a powerful net spin. Many spins aligned together form domains [4]. Although each of these domains has a net magnetization, neighboring domains will tend to align anti-parallel to each other to minimize energy. In thin films, in the absence of an external magnetic field, these domains form intricate labyrinth-like patterns as represented in Fig. 1.1, so that the net magnetization is zero. If the film is defect-free, these patterns are completely random.

Ferromagnetic materials are known best for their profoundly nonlinear responses to an applied magnetic field, which allows them to form permanent magnets. The magnetization of a ferromagnetic material is dependent not only on the applied field, as in diamagnetic or paramagnetic materials, but also upon the history of the material. Its response to a full field cycle is thus known as ferromagnetic hysteresis. Bulk measurements of net magnetization can be conducted with a variety of magnetometry techniques. We have used BYU's recently acquired vibrating sample magnetometer (VSM) for all of our magnetization measurements.

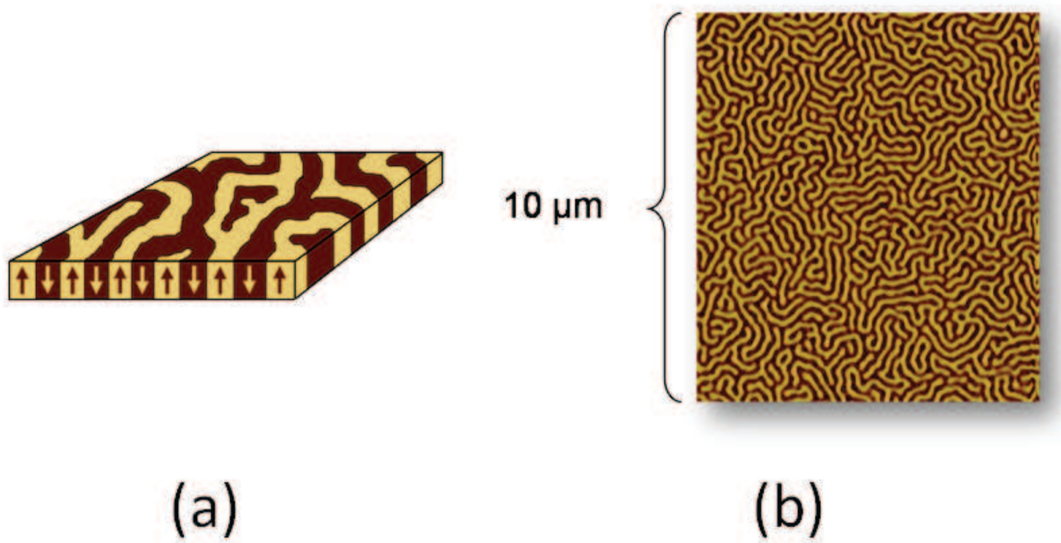

Figure 1.1 Ferromagnetic Domains (a) In Ferromagnetic Materials, the spins tend to form large regions of like spin called domains. In our films, these domains are typically on the order of hundreds of nanometers and tend to prefer perpendicular magnetization. (b) A typical ferromagnetic domain pattern in a perpendicularly magnetized thin film. This image is ten microns by ten microns, and was obtained by magnetic force microscopy at BYU (courtesy Andrew Westover). Because there was no external field during demagnetization of this sample, the domain pattern is isotropic. Although the direction of the domains are random, they exhibit relatively constant periodicity in all directions.

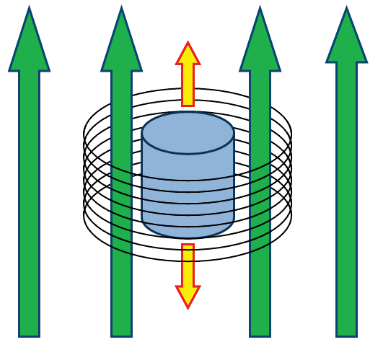

Figure 1.2 Vibrating Sample Magnetometer The functional components of a vibrating sample magnetometer (VSM). A sample, shown in blue, is placed within detecting coils, shown in black, and vibrates vertically, depicted by yellow arrows. The vibration of the sample induces a voltage in the detection coils that is proportional to its magnetic moment. The entire apparatus is placed inside of a superconducting solenoid, whose field is parallel to the direction of vibration, shown in green. Thus the response of the sample's magnetization to applied field can be measured.

The VSM technique utilizes Faraday's law of induction to measure magnetization. The sample is placed inside of a coil of wire and vibrates up and down (in and out of the coil), as depicted in Fig. 1.2. The changing magnetic flux caused by this vibration induces a sinusoidal voltage in the wire, whose amplitude can be measured very accurately. Because a constant applied field will not change the voltage produced, magnetization can be measured with an *in-situ* applied field.

An example of a VSM measurement on our [Co/Pd]/IrMn system is shown in Fig. 1.3. In this measurement, we can observe four principle features unique to ferromagnetic materials, labelled with letters in the figure. Beginning at the star, where the system is saturated in the positive direction, as the external field is decreased, the sample follows the descending branch indicated by the green arrow. Point A corresponds to nucleation, where domains antiparallel to the field first begin to form.

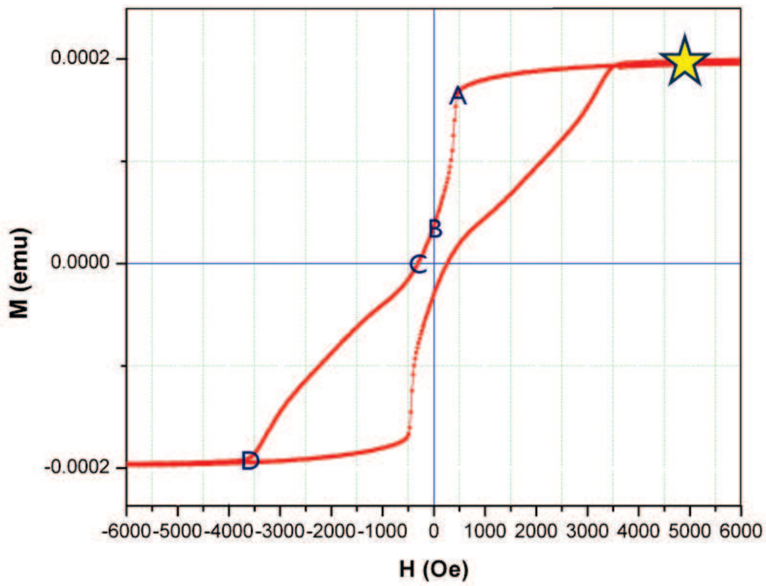

Figure 1.3 Ferromagnetic Hysteresis Ferromagnetic hysteresis loop of our [Co/Pd]/IrMn sample at room temperature measured using VSM. The magnetization of a ferromagnetic sample is dependent not only on the applied field, but also on the history of the sample. Beginning at the star, the sample follows the descending branch, indicated by the green arrow, to points A, nucleation; B, remnance; C, coercive point; and D, saturation. The ascending branch (not labelled), is the mirror image of the descending branch.

At point B, the external field is zero, but there is still some magnetization; this is called remnance, and is what makes permanent magnetization possible. As the field continues to decrease (now in the reverse direction), the sample reaches the coercive point (point C), at which there are equal domains parallel and antiparallel to the field, and thus no net magnetization. Finally, the field saturates the sample at point D, so that the entire sample is magnetized parallel to the reversed field. From this point, if we increased the field back to its original value at the star, the sample would follow the nonlabelled ascending branch of the hysteresis loop, as indicated by the arrow. These two branches are symmetrical.

Because our films are very smooth (meaning that they have very few structural defects), at saturation, they are magnetically uniform. In the absence of defects, the location of nucleating domains is mostly random. Therefore, in such films, all previous domain patterns are lost just as data coding regions are lost when a strong magnet comes near a hard disk. Thus, every time a sample is cycled through its hysteresis loop, the domain pattern is different. However, it has been found that if a film presents some structural defects, the formation of domain patterns is less random; in fact, the greater the structural disorder of a sample, the less the domain patterns change from cycle to cycle. This tendency to "remember" a previous domain pattern has been termed disorder-induced magnetic domain memory [5]. In a film with very high memory, data stored in magnetic domains could not be erased by an external field.

1.3 Exchange-Bias Films

Structural defects like large grains produce relatively low memory, and are difficult to control. For this reason, films were developed in which antiferromagnetic (AFM) layers are interspersed between ferromagnetic (FM) layers (see Fig. 1.4).

Anti-ferromagnetic materials are similar to ferromagnetic materials in that they are composed of particles with intrinsic spin. In ferromagnetic materials, neighboring spins tend to align parallel with one another, while in antiferromagnetic materials these spins tend to anti-align with one another, creating a net spin of zero in the bulk of the material. However, on the outer layers of particles, uncompensated spins form which are not cancelled out locally. Below a certain temperature, called the blocking temperature, or T_B, these uncompensated spins interact strongly with the spins of ferromagnetic atoms near the interface between the two materials [6], [7]. This interaction is referred to as exchange-coupling [8]. Above T_B, the uncompensated

Figure 1.4 Exchange-bias Multilayer An example of an exchange-bias multilayer. In our $[[Co(4\mathring{A})/Pd(7\mathring{A})]_{12}IrMn(24\mathring{A})]_4$ films, an antiferromagnetic Ir/Mn alloy layer is periodically inserted into a repeating ferromagnetic Co/Pd multilayer. Exchange coupling occurs at the interface between the ferromagnetic and antiferromagnetic layers, causing uncompensated spins in the antiferromagnetic material very near the interface.

spins tend to align with the spins in the neighboring FM layer. Below T_B, however, the AFM spins are frozen in place and are affected little by external fields. Thus, when the sample is cooled, the domain pattern in the FM layer is imprinted into the AFM layer, and will not change with external fields.

A simple test to measure this exchange-coupling interaction is to cool the sample below T_B in the presence of a saturating external field. Indeed, this is the way exchange-coupling was first observed [8]. In this case, called field cooling (FC), the domain configuration to be imprinted in the AFM material is uniform in the direction of the external field, so that when measurements are taken below the blocking temperature, the external field felt by the FM layer is modified by the saturated AFM layer. This field bias causes a horizontal shift in the sample's hysteresis loop, as shown in Fig. 1.5. Thus, films which display this kind of interaction are called exchange-bias (EB) films.

Slightly more complex is the zero-field cooling (ZFC) state. Here, domains in

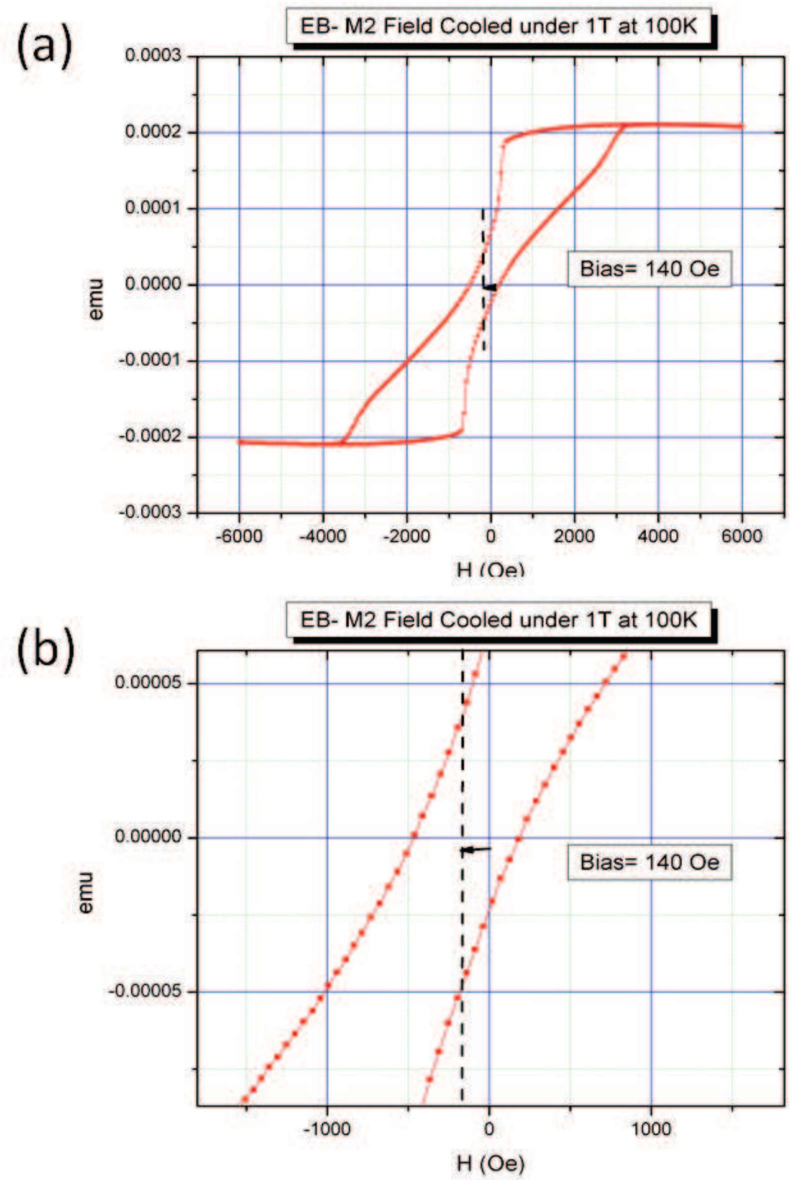

Figure 1.5 Field-Cooled Exchange-Bias (a) The major hysteresis loop following field cooling. (b) a zoomed-in region of the loop shown above, making the horizontal bias more clear. The sample was cooled in the presence of a saturating magnetic field of 10 000 Oe (1T), and the hysteresis loop was measured at 100K. The entire hysteresis loop has shifted to the left by 140 Oe (14 mT) because uncompensated spins in the saturated antiferromagnetic layers modify the effective field experienced by the ferromagnetic layers. Lower temperatures would enhance the EB by decreasing thermodynamic effects, and a greater bias would be observed at lower temperatures.

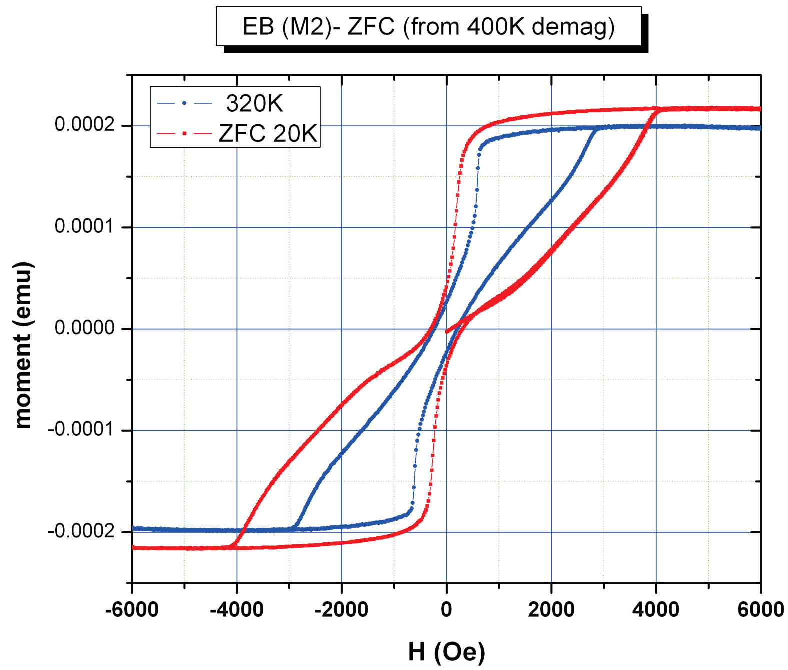

Figure 1.6 Zero-Field Cooled Exchange-Bias Zero-field cooled (ZFC) magnetization loop (red), shown with a loop taken above T_B (blue). Exchange-coupling causes a plateau in the magnetization after the coercive point is reached, as well as an increased saturation point.

the FM layer imprint onto the AFM layer as the sample is cooled. Because of the interaction between the FM and AFM layers, the sample's FM spins try to remain aligned with the uncompensated spins in the AFM layer as long as possible to minimize energy. This causes a plateau in the hysteresis loop near the coercive point. This also increases the saturation point of the sample. Then, as the field is reversed, when new domains nucleate after saturation, they will continue to align with the uncompensated spins in the AFM layer. Thus, below T_B, the pattern formed by the domains will ideally be the same each field cycle.

We have studied a $[[Co(4\mathring{A})/Pd(7\mathring{A})]_{12}IrMn(24\mathring{A})]_4$ multilayer made at Hitachi [9]. In these films, the thickness of Co and Pd have been optimized to obtain the best EB effect in the direction perpendicular to the film [10]. We measure the magnetic domain memory of this system as a function of temperature, applied field, repeated field cycling, and spatial scale.

Chapter 2

X-ray Correlation Technique

2.1 X-Ray Resonant Magnetic Scattering (XRMS)

X-ray resonant magnetic scattering (XRMS) is a recently developed advanced technique to investigate microscopic domain morphologies [11]. Our XRMS measurements were performed by K. Chesnel [1] at the Advanced Light Source at Lawrence Berkeley National Labs, Ca. In our experimental setup, illustrated in Fig. 2.1, a sample is placed in a variable *in-situ* magnetic field (H) and irradiated by spatially coherent x-rays tuned to the magnetic resonance energy [12]. When the x rays irradiate the sample, they undergo Bragg-like scattering and the signal is recorded on a CCD camera. A blocker is also used to prevent saturation and damage of the detector by direct light.

This scattering process is analogous to a Fourier transform of the domain pattern in the sample, but because we are only able to detect intensities, the phase of the transform is lost [13]. This makes manual reconstruction of the domain pattern itself very difficult. Scattering patterns live in the reciprocal space (quantified by the vector Q, or the distance to the origin on the CCD image). Qualitatively, a peak at

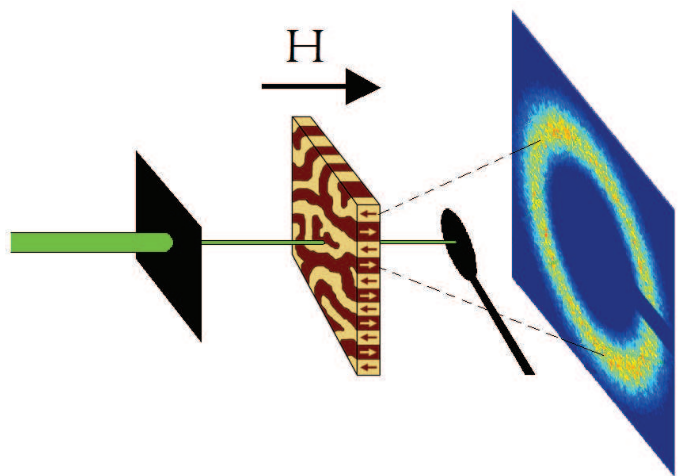

Figure 2.1 X-ray Resonant Magnetic Scattering Experimental Setup In our X-Ray resonant magnetic scattering (XRMS) setup, temporally coherent x-rays from a synchrotron source were tuned to the magnetic resonance frequency of the sample by a monochromater, and made spatially coherent by use of a 20 µm pinhole (spatial filter). These coherent x rays illuminated the sample (not shown to scale), in the presence of an applied field (H) and produced Bragg-like scattering in the far field, which was detected by a CCD camera. The pattern produced is unique to the domain pattern irradiated by the x-rays. Our isotropic domain patterns create a ring shape in the scattering space A blocker was used to prevent saturation and destruction of the CCD by direct light.

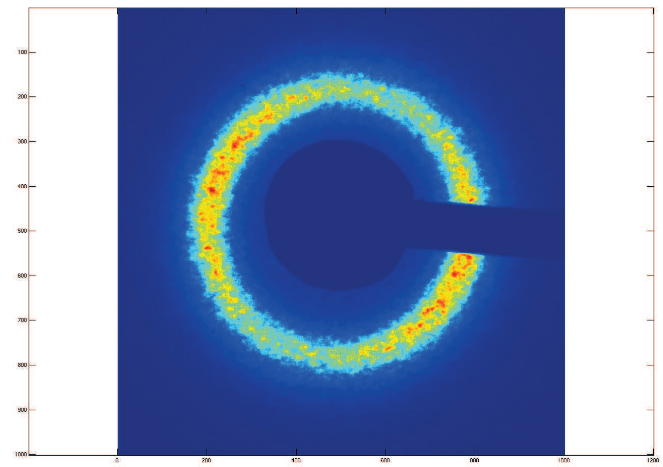

Figure 2.2 Typical XRMS Image An XRMS image taken after demagnetization at zero field. The ring shape indicates an isotropic domain pattern. The average periodicity of the domain pattern is given by the radius of this ring, and the coherence length of the pattern by its width. Here, the radius of the ring is about 310 pixels, which corresponds to an average periodicity of about 380 nm.

a distance Q from the origin indicates a periodic signal whose periodicity is inversely proportional to Q, and the direction of this periodic signal is given by the angular position of the peak. The scattering images contain more than one kind of scattering. Four main elements are present in every image, seen in Fig. 2.2: high-intensity charge scattering (both coherent and incoherent), concentrated at the center of the image (mostly blocked by the blocker); incoherent magnetic scattering, which forms the disc or ring shaped envelope; coherent magnetic scattering, which produces speckles [14]; and low-intensity noise.

The charge scattering will not change at all throughout the hysteresis loop, because the structure of the film does not change during magnetization, but rather the orientation of the spins within the lattice. It will only change, and then only marginally, with a change in temperature.

The incoherent magnetic scattering produces the disk or ring, and the radius of this ring is a measure of the long-range periodicity of the domain pattern. In the example shown, the radius of the ring is about 310 pixels, which corresponds to a periodicity of about 380 nm in the real space. This long-range periodicity is related to the net magnetization of the film. Thus, at a given point of the hysteresis loop, the incoherent magnetic scattering will be the same. The width of the ring is also instructive, and is related to the correlation length of the domain periodicity, or the average distance over which domains tend to be aligned parallel with one another.

The coherent magnetic scattering, on the other hand, which produces the speckle pattern, relates to the short-range disorder of the domains, i.e. the exact shape of the domains themselves. Thus, although the incoherent scattering is instructive regarding long-range periodicity, the speckles (and only the speckles) are unique to the exact shape of the domains in the irradiated portion of the sample. The speckle pattern, then, can be thought of as the fingerprint of the domain pattern.

2.2 Speckle Isolation

The proportion of incoherent scattering to coherent speckle scattering is an indication of the spatial coherence of the beam. In our case, coherence ranged from about 10-20% [15], depending on the quality of the incident light, so that the intensity of the incoherent scattering is much greater than that of the speckle pattern. Thus, any quantification of the similarity between scattering patterns will be dominated by the incoherent scattering, which does not change from cycle to cycle at a given point in the hysteresis loop. However, this incoherent scattering may be manually removed to leave the pure speckle behind (see Fig. 2.3).

Our first isolation step utilizes scattering images taken with an applied field higher

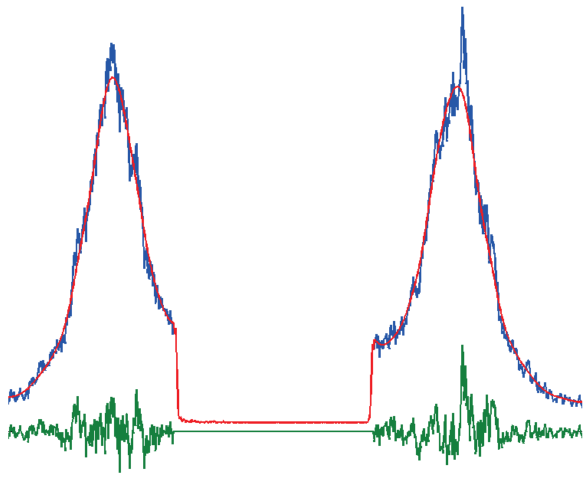

Figure 2.3 Speckle Isolation A vertical slice of the scattering image shown in Fig. 2.2. The blue line shows the image before speckle isolation. The red line is a result of smoothing the blue line, approximating the scattering that would result if the light were completely incoherent. This is subtracted from the original image to isolate the pure speckle, shown in green.

than the sample's saturation field. In these images, all domains in the sample have saturated in one direction, so the magnetic portion of the scattering is concentrated about Q=0, behind the blocker. Using these as reference images, we can remove much of the non-magnetic signal from the rest of our images. However, because the intensity of x ray source was not constant over the course of the experiment, this does not eliminate all of this non-magnetic signal.

Next, we can manually remove regions of the image behind the blocker that are affected by charge scattering. This charge scattering will change each time the CCD, blocker, or sample are moved between experiments. These regions are basically zeroed out.

Our final task in isolating the pure speckle signal is the removal of the incoherent magnetic scattering. Because this signal does not change for a given field value, we do not want to include it in our correlation calculation. A useful algorithm was written

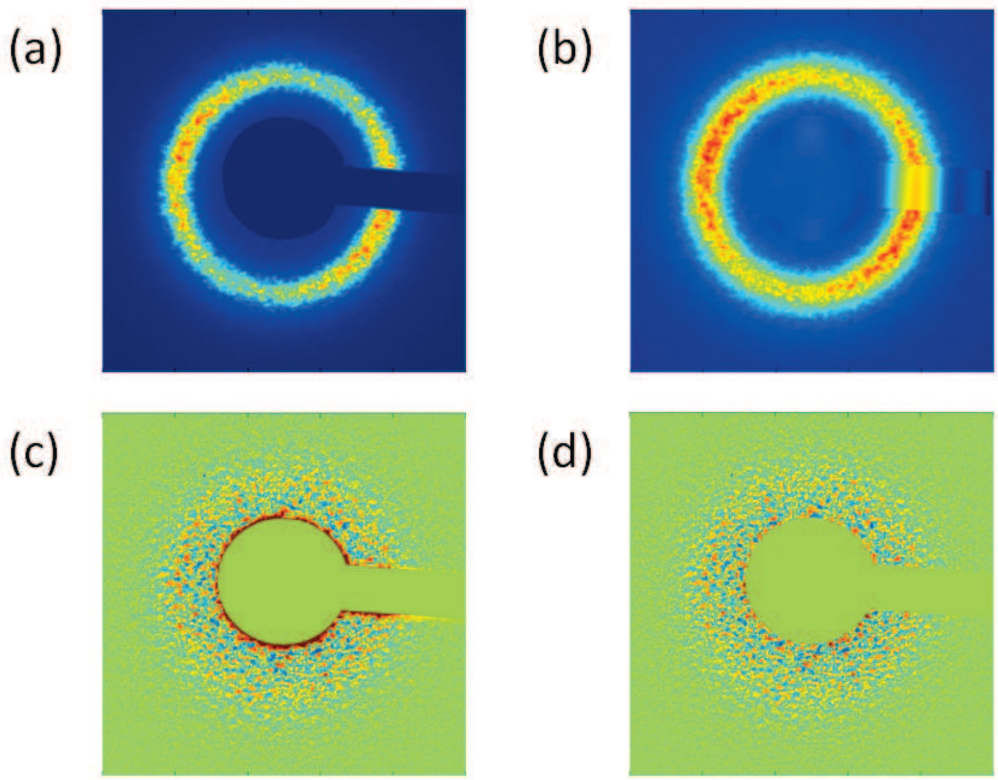

Figure 2.4 Blocker Fitting (a) A raw scattering image as detected. (b) The same scattering image after fitting the regions behind the blocker to the envelope. This is now ready to smooth to fit the incoherent envelope more accurately than the image with the blocker. Using this method, we were able to improve the resulting speckle patterns, seen in (c) without blocker correction, and (d) with blocker correction. This enables us to use more of the image, which results in more statistics and a better ability to analyze data near the center of the image.

by Brian Wilcken to successively smooth the image with a small averaging box until all speckles were removed to obtain the approximate incoherent envelope (for a complete description see [15]). This envelope was then subtracted from the image, ideally leaving behind pure speckle, as shown in Fig. 2.3. However, because of the artificially low region created by the blocker, this averaging program underestimates the envelope near the blocker, which results in unusable regions in the pure speckle image near the blocker. Brian's solution was to eliminate these areas from the calculation, but in order to have the best statistics possible and extend Q-selective measurements (discussed in the next section) near the center of the image, these areas needed to be included.

This problem occurred because averaging the values of the envelope with the low values behind the blocker brought the envelope estimation down near the blocker. My solution was to fit the values behind the blocker with two independent orthogonal 1-dimensional polynomial fits prior to smoothing. This fit is shown in Fig. 2.4(b). When smoothing is performed, the envelope is always surrounded by regions of the same magnitude, eliminating the artificial boundary created by the blocker. The smoothing fit is performed, and blocker region is then removed again so that our blocker fit never directly becomes part of the results. This seems to allow for greater statistics and lower radius measurements, as seen in Fig. 2.4(d)

We are now ready to quantify the similarity between domain patterns by comparing their respective speckle patterns.

2.3 Speckle Cross-Correlation

In order to quantify the similarity between two speckle patterns, we have used cross-correlation metrology. In one dimension, the cross-correlation between two functions, a(x) and b(x), shifted by $\Delta x = n$, is defined by

$$[a \otimes b](n) = \int_{-\infty}^{\infty} a^*(x) b(x+n) dx,$$

where * indicates the complex conjugate. For our special case of two-dimensional, real, discrete images, the cross-correlation becomes

$$[A \otimes B](n,m) = \sum_x \sum_y A(x,y) B(x+n, y+m),$$

where each sum is over all values of x and y in the frame of the images, and the complex conjugate has been left off because the images are real. Physically, this means that two images are superimposed upon one another (with a horizontal shift n and a vertical shift m), the overlapping pixels are all multiplied together, and the products are summed up to give the cross-correlation at that shift. This is then repeated for all possible shift values, creating a matrix of cross-correlation results as a function of shift. If A and B are very similar, a peak will form around the zero shift value (n=m=0) because essentially every value is being squared (See Fig. 2.5). This is especially prominent if the intensity in A and B is centered about zero so that there are positive and negative values. The more similar the two images are, the higher the integration of the correlation will be.

As a side note, since we are dealing with images that each have about one million pixels, this result is very useful, but also extremely computationally expensive (the complexity is of order N^N). However, we can use Fourier Transforms to speed up the process because

$$F([A \otimes B]) = F^*(A) \times F(B),$$

where F indicates the Fourier transform. If we were to calculate the cross-correlation by this method, the complexity would change from N^N to $\log N$, and our computations become orders of magnitude simpler.

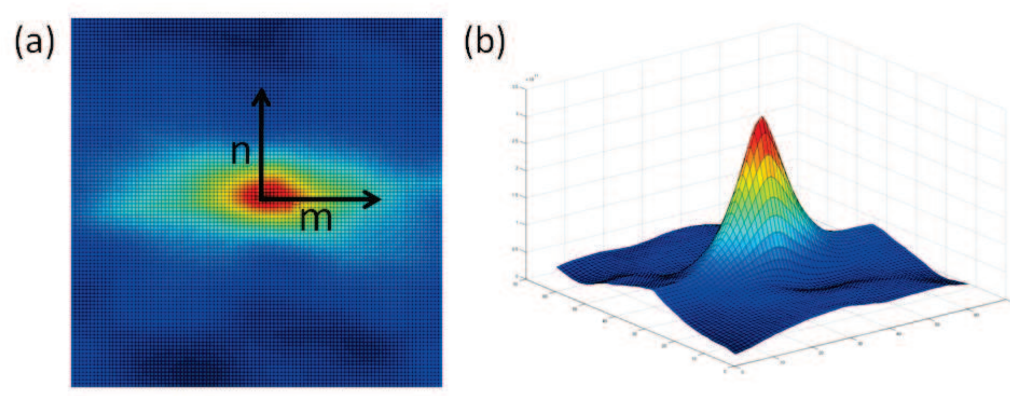

Figure 2.5 Cross-correlation A typical cross-correlation result. The maximum of the peak is at the zero-displacement region (n=m=0), indicating positive correlation between the images it was taken from. (a) is a two-dimensional view of the correlation, while in (b) it is visualized three-dimensionally.

Since this correlation result is highly dependent upon the average intensity of each image, it is difficult to compare correlation results. For example, two high-intensity, very different images may produce a higher cross-correlation result than two extremely similar low-intensity images. To solve this problem, we define a normalized value ρ [16]

$$\rho \equiv \frac{\sum[A \otimes B]}{\sqrt{\sum[A \otimes A]\sum[B \otimes B]}},$$

where the sum is carried out under the peak. The factors in the denominator are called autocorrelations, and are simply the cross-correlation of an image with itself. Clearly, if B and A are exactly the same image, ρ will have a value of 1. If there is no correlation whatsoever, ρ will be near zero. Thus, ρ provides an intuitive, quantitative measure of the similarity between two images.

We have used return-point memory (RPM) as our measure of the memory of our system. In RPM, correlations are performed between an image taken at a given point of the hysteresis loop and an image taken after returning to the same point after a (or multiple) cycle(s) through the full loop. Thus, if an image were taken at point C of

Fig. 1.3, the RPM would be given by the value of ρ between this image and an image taken after going to point D, up to the star, and back to point C again. Thus, we are quantifying the similarity between the domain patterns when the net magnetization is exactly the same.

2.4 Q-selective analysis

2.4.1 Reciprocal Space

So far, we have only discussed correlating whole images with one another, allowing for 'global' statistical domain memory to be measured; however, understanding the inverse relationship between spatial scales in the scattering space and the real space allows us to extract spatial information as well. For example, a speckle spot near the center of the scattering contains information about periodicity of domains on a larger scale than a speckle farther from the center of the scattering. For this reason, the scattering is said to be in reciprocal space.

Generally, Q represents the scattering vector from the origin, or center of the scattering (where undeflected light would hit the screen), and the region of interest. Small Q then corresponds to a large spatial scale in the real space, while large Q correspond to a small spatial scale. In our scattering geometry, the light is scattered in transmission, as shown in Fig. 2.6. Therefore, the relationship between Q and distance in the real space is given by

$$Q = \frac{2\pi}{p} = \frac{2\pi}{\lambda}\sin\theta$$

,

where p represents the periodicity of domains in the real space, λ represents the

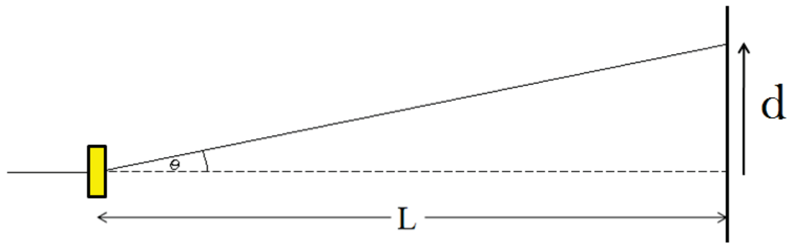

Figure 2.6 Transmission Geometry The scattering vector (Q) can be determined from the scattering angle θ and the wavelength of light. Theta is experimentally measured by d, the distance between the scattering feature and the center of the scattering on the detector, and L, the distance from the sample to the detector. We are dealing with very small angles, so $theta \approx d/L$.

wavelength of the radiation (1.6 nm in this study), and θ is the scattering angle. This is equivalent to Bragg's law in transmission. Experimentally, the angle θ is determined by the small angle approximation, $\theta \approx d/L$. This, combined with the definition of Q, gives us the useful relationship

$$p = \frac{\lambda}{sin\theta} \approx \frac{\lambda L}{d}$$

.

Because Q is related to d by a simple proportionality constant $2\pi/\lambda L$, we will simply refer to the distance d as Q in the remainder of this thesis.

Because the distance from the center of the image indicates a spatial scale, if we select only speckles within a given range of Q from each image and perform correlations between these isolated rings (see Fig. 2.7), we could measure the memory at the spatial scale indicated by the Q-vector. By iterating over all values of Q, we can determine the spatial dependency of memory.

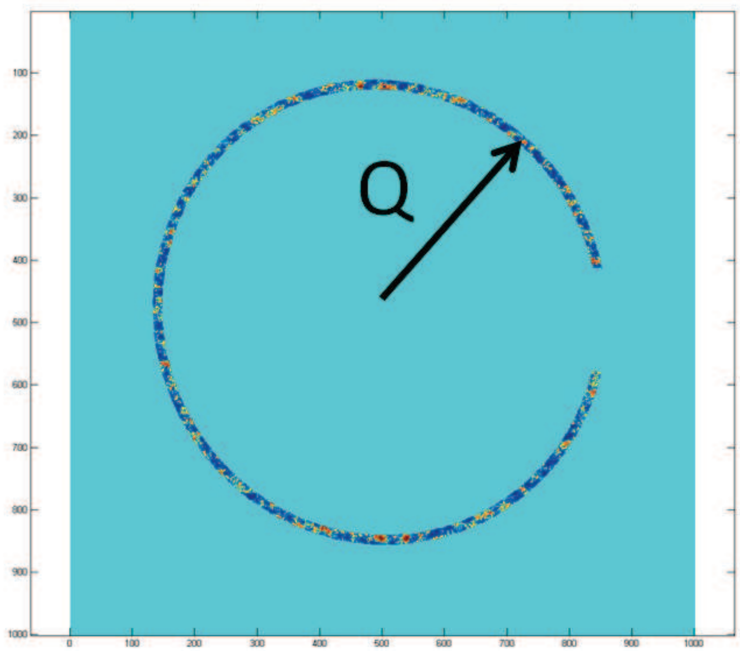

Figure 2.7 Isolated Ring One isolated ring from the speckle pattern. Correlations of isolated rings is specific to the spatial scale given by the radius of the ring (Q).

2.4.2 Q-selective Method

To calculate Q, we must first know where the origin of the scattering is; the center of the scattering is not necessarily the central pixel of each scattering image. To find the center of each image, I smoothed several representative images until no hint of speckle remained, then fit this to an ellipse. In general, ellipses with very slight eccentricity fit the data better than circles, probably because there was a slight tilt to the CCD camera when recording the data. Once the center of the scattering is determined, the next task is to choose an appropriate width of each ring ΔQ, which will determine our ultimate spatial resolution. The smaller ΔQ is, the better our spatial resolution will be, but the statistics of each point will be less. Thus, there is a trade-off between spatial resolution and statistical reliability. In fact, our choice of normalization using autocorrelation has a subtle dependence on the size of the area correlated that we have not yet taken into account.

Our coefficient of memory, ρ, was normalized by dividing each cross-correlation by the square root of the autocorrelations of each image. Our scattering data, like all measurements, contains random noise. Because this noise is random, it will be different for each image, and will tend to cancel itself out of all cross-correlation results. However, when an image is correlated with itself, as in autocorrelation, the noise is squared at zero shift, producing a sharp peak in the autocorrelation at the zero displacement pixel (ZDP)(see Fig. 2.8). This peak becomes much sharper at larger Q-values. It is difficult to estimate the noise level in our images, so we do not know how much of the signal at zero shift is a result of similarity between speckle patterns and how much is a result of noise. As ΔQ becomes small, the autocorrelation peaks become sharper, and the peak integration is increasingly dominated by the ZDP (See Fig. 2.10).

Three possible corrections were considered to eliminate this noise, as shown in Fig. 2.9. First, we could simply remove the ZDP entirely. This would eliminate the uncertainty in the contribution of the noise, but makes the very wrong assumption that the signal at the ZDP is entirely due to noise. As a more sophisticated approach, we can fit the peak, with the ZDP removed, to a Gaussian or Lorentzian shape, and replace the central pixel with the fitted value. This would have much less of our real signal removed, but at high Q, where the signal is much lower and the peaks are much sharper, the fit tends to overestimate the central pixel by orders of magnitude, effectively zeroing out ρ in several regions of otherwise usable data. Also, it will make the cross-correlation algorithm much more complicated, and because the fitted data point may be higher or lower than the actual signal at that point, we would not know whether we are underestimating or overestimating our value of ρ. As a compromise between these two approaches, a nearest-neighbor fit on the ZDP has been adopted,

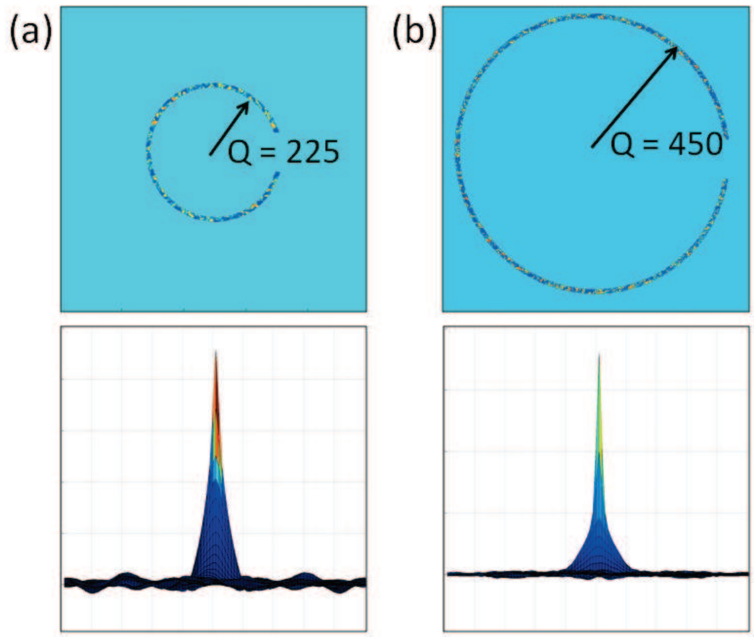

Figure 2.8 Narrowing Autocorrelation Peaks Two autocorrelation results of peaks at (a) Q = 225 pixels and (b) Q = 450 pixels. The autocorrelation peaks become sharper, and thus more influenced by the central pixel, as the radius increases due to a decreased signal-to-noise ratio.

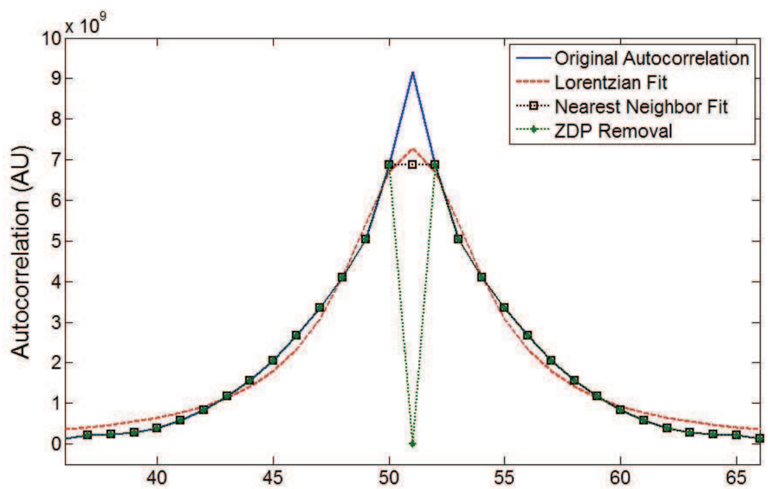

Figure 2.9 Autocorrelation ZDP Correction Options Three possible corrections for the zero displacement pixel. (Red) The data is fit to a lorentzian shape with the central pixel removed, and the fitted value is used to replace the ZDP. (Green) The nearest neighboring point to the ZDP is used to approximate the ZDP. (Black dotted) The ZDP is simply removed.

in which the central pixel is approximated by the highest neighboring value. This simple approach is computationally cheap, keeps most of the value of the central pixel, and always slightly underestimates the autocorrelation, and because ρ divides by this autocorrelation, we obtain an upper estimate on the value of ρ.

We have chosen to use the neighbor fit corrected data in conjunction with the uncorrected data to form an upper and lower limit for ρ. Because we are dividing by the autocorrelation, an overestimation in autocorrelation results in an underestimation in ρ, while an underestimation in the autocorrelation causes an overestimation of ρ. Thus, with both an underestimation (no ZDP correction) and an overestimation (neighbor-fitted ZDP) of ρ, we obtain absolute upper and lower limits of ρ, and have a better idea of the uncertainty at each point due to the ambiguity of the ZDP signal. A typical example is shown in Fig. 2.11.

As can be seen from the figure, the difference between the corrected and uncorrected values is very small compared to the other features of the graph. I therefore will report only the nearest-neighbor corrected values in the remainder of this thesis.

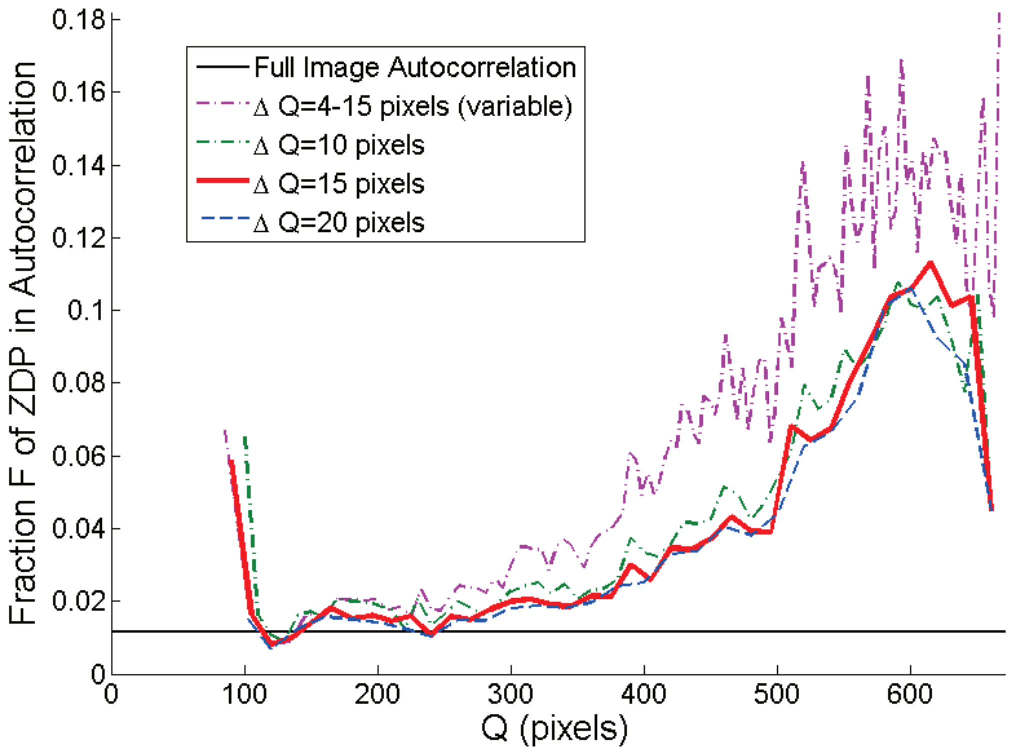

Figure 2.10 Fraction of ZDP in Autocorrelation Fraction F of the ZDP contribution to the autocorrelation signal in an image taken after demagnetization at 30K, H=0. F is plotted as a function of radius from the center of the scattering (Q) for several different values of ΔQ, as well as for the whole-image autocorrelation. As Q increases, F also increases. This is due to the width of the ring truncating the width of the autocorrelation peak. An increase from 4 to 10, and 10 pixels to 15 pixels provide significant reductions in F, and thus the role played by the ZDP, but after 15 pixels the decrease in F is counterbalanced by the loss in resolution in Q. We have therefore chosen to use a constant 15 pixels as ΔQ in our analysis.

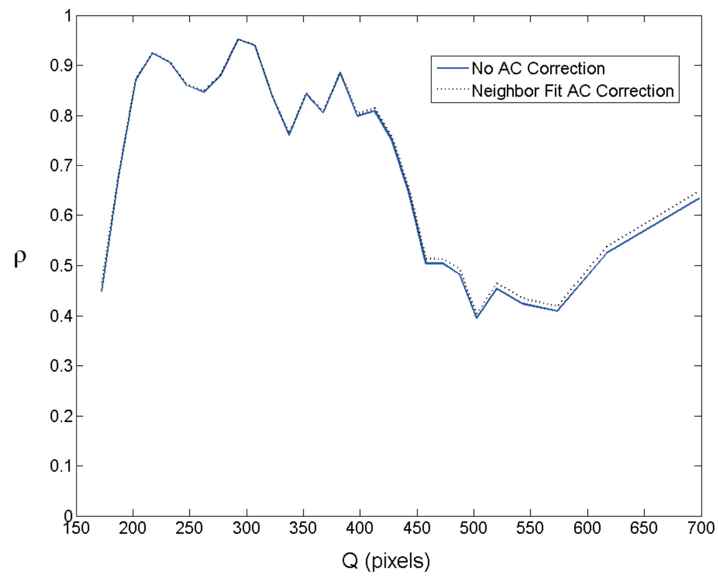

Figure 2.11 ZDP Uncertainty A plot of ρ as a function of Q in the coercive region at T=175K. The upper and lower limits of ρ due to uncertainty in the zero-displacement pixel are indicated by an uncorrected value of ρ giving the lower limit (blue), and the nearest-neighbor correction of the autocorrelation (AC) giving the upper limit (black dotted).

Chapter 3

Results and Discussion

3.1 "Global" Domain Memory Correlations

We first looked at the "global" degree of correlation by cross-correlating full images. We were able to quantify the memory ρ as a function of magnetic field at different temperatures, as shown in Fig. 3.1. The speckle images used here (and throughout this thesis) were obtained after performing demagnetization above 300K, followed by zero-field cooling down to 20K, and then up to the final indicated temperature. We followed the major hysteresis loop, collecting scattering images at several key field values. In general, three full major loops were imaged at each temperature. This process was repeated at 30K, 60K, 120K, 175K, 220K, which are below T_B, and 335K, which is above T_B, to see how the temperature affected the ability of the frozen AFM layer to act as a template for domain nucleation.

The memory below T_B has a very reproducible shape as a function of H. The field begins at positive saturation, and follows the descending branch of the major hysteresis loop. At nucleation, when domains begin to form along the descending loop, the memory is low, then as we continue to decrease the field, the memory increases to

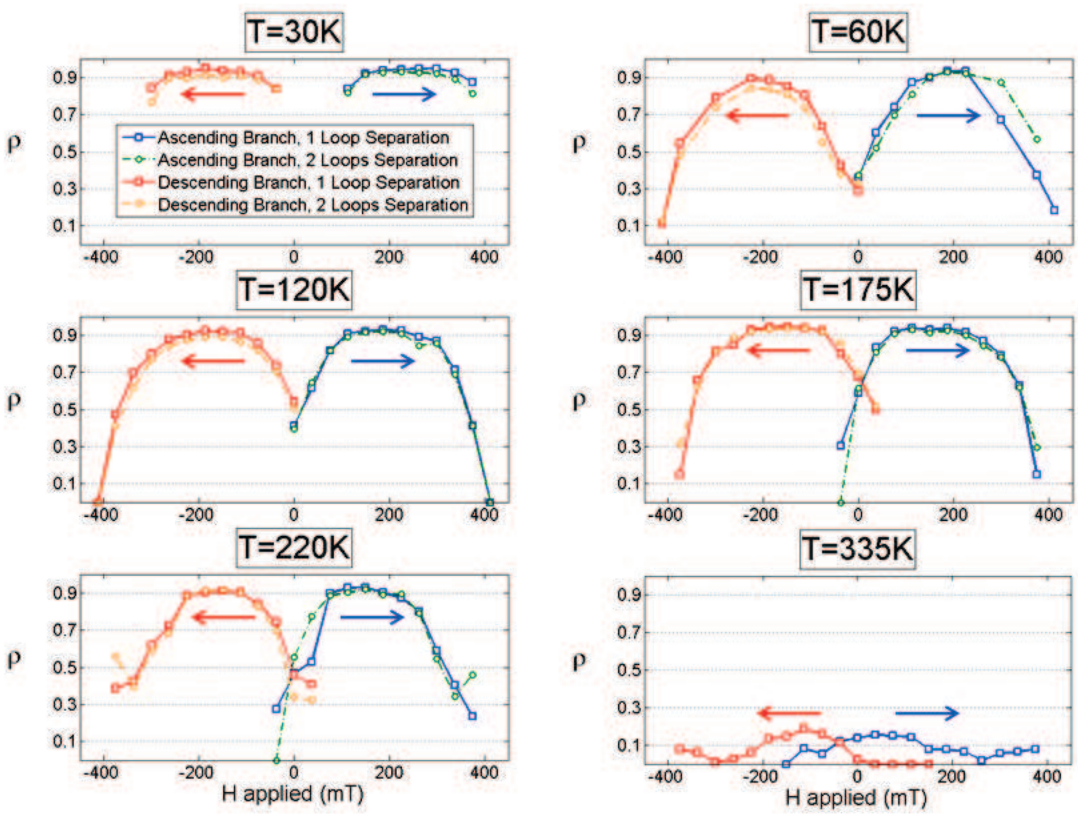

Figure 3.1 Whole-Image Correlations Magnetic domain memory at 30K, 60K, 120K, 175K, 220K, and 335K. The blocking temperature T_B of the AFM layer is 275K, and when below this temperature, the memory is very high. The general response of the memory to applied field is very consistent throughout this temperature range. At nucleation and saturation fields, the memory is very low, increasing to a plateau in the coercive region. Above T_B, the correlation measured is much lower than at cooler temperatures. In addition, because our correlation algorithm becomes much less accurate for low memory measurements, the actual memory in the system at high temperature may be much lower than shown.

a maximum value about the coercive field, which is always over 0.9 when below T_B. As the field continues to move along the descending branch toward saturation, the correlation decreases. The ascending branch exhibits a symmetrical behavior. Above T_B, however, we see a different shape and much lower maximum in the memory[1]. This behavior can be explained by the specific memory mechanism used in these samples.

After cooling below the blocking temperature, T_B, the domain pattern that was present in the sample when it was cooled is frozen in the AFM layer at the FM/AFM interface. The uncompensated spins in the AFM then are able to act as a template for domain formation for the FM layer because it is energetically favorable for the FM spins at the boundary to align with the very small field created by the template. This exchange coupling is what makes such high memory possible.

Although domains will always lie along the AFM template, during nucleation, there are many energetically equivalent locations for nucleation, as shown in Fig. 3.2. Because of this, the nucleation of domains is very random at first. Then, as more area is filled with domains, there is much more of a chance for domains to be in the same location if they lie upon the template. This is why very high memory is observed, about 0.95, near the coercive point. As these domains widen during the propagation phase, and the number of domains decreases, the memory stays strong and eventually decreases at higher fields because the domains likely disappear in a random way, just as they nucleate in a random way. This plateau in the memory

[1] Although our correlation program is good at quantifying memory in highly correlated images, when correlation is low it becomes less accurate. This high temperature series, for example, had a single pronounced peak in the autocorrelation, but no clear central peak in the cross-correlation. However, because a good deal of signal was present in the cross-correlation space that was being interpreted as a central cross-correlation peak, we decreased the size of our peak integration area for both autocorrelation and cross-correlation in this series to show better how low the correlation actually is.

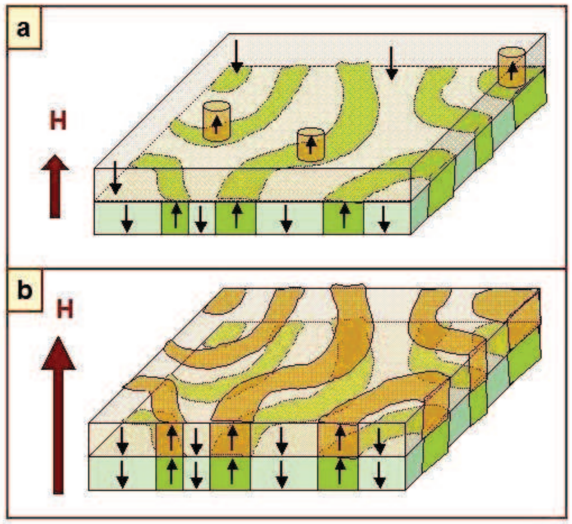

Figure 3.2 Exchange-bias Nucleation Sketch of magnetic domains nucleating (a) and propagating toward the coercive point (b). In both sketches, the FM layer lays on top and the AFM lays below. From Karine Chesnel et al. [1]

corresponds to the same plateau seen in the hysteresis of this system in Fig. 1.6. Finally, at saturation, there are no domains left, and no signal left in the scattering, so no memory is observed.

We hoped that the memory of our samples would decrease very little as temperature was increased towards T_B around 275K. When we plot the maximum in the memory at each temperature, it seems to stay very robust as temperature increases, and remains over 0.90 at 220K (see Fig. 3.3). The significant decrease in memory at 335K, which does not follow the slight decreasing trend of the memory at lower temperatures, is evidence that there is a phase transition. Thus, the high memory we observe below T_B is indeed induced by this exchange-coupling interaction.

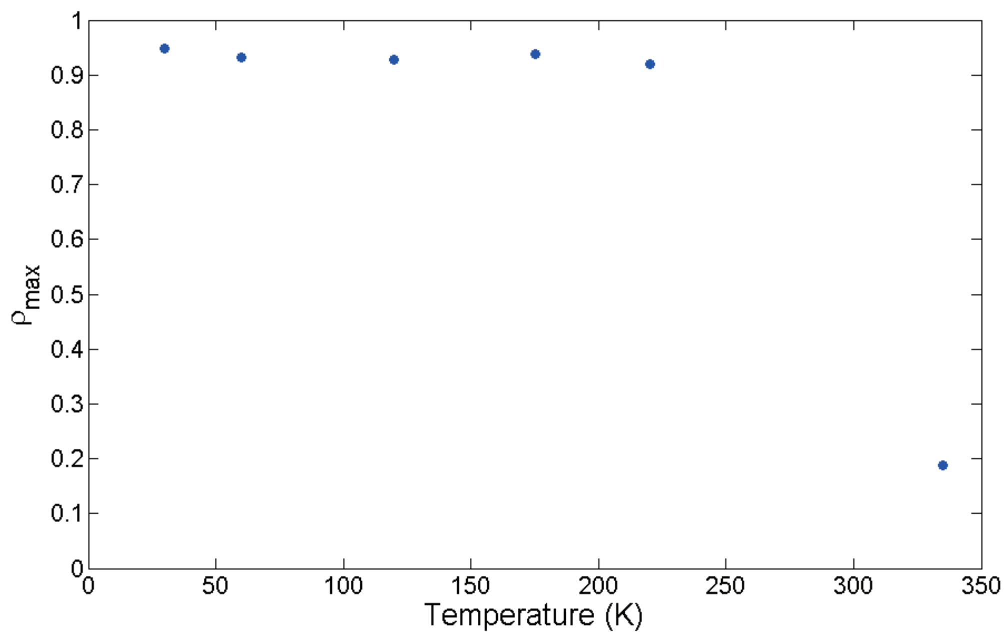

Figure 3.3 Temperature Dependence The maximum correlation value ρ at each temperature. From 30K to 220K, the maximum memory of the system is over 0.9, and remains very high throughout this temperature range. Above T_B, the memory drops to about 0.19.

3.2 Q-selective Correlation

This same data set was analyzed using a Q-selective approach. In this procedure, each line plotted previously becomes a 2-dimensional (Q,H) map detailing the dependence of memory on spatial scale (Q) as well as field (H) as seen in Figs. 3.4, 3.5, 3.6, and 3.7. Memory in these maps follows the same general field dependence that we saw previously, but also contains several important features that are spatially dependent.

We have plotted four maps for each sub-T_B temperature, corresponding to images separated by one full hysteresis loop and two full hysteresis loops, for the ascending and descending branches respectively. These all have a similar overall appearance. We also plot our results for 335K, above T_B, and note that it has a very different appearance.

To better understand the features present in the low-temperature maps, we will first compare our correlation map with a map of the intensity of the speckle signal at the same H and Q values, as shown in Fig. 3.8. These patterns have three main differences: first, the overall shape of the intensity in(Q,H) space is very different than that of the memory. Starting from the bottom left corner of the map, the intensity seems to follow a general up and to the right directivity along the first diagonal. In other words, when the field increases, the Q at which the maximum intensity occurs also increases. This means that the domain periodicity progressively decreases to reach a minimum value at about 400 nm. The memory, on the other hand, appears to follow a down-right directivity, where an increase in applied field is accompanied by a decrease in the Q at which correlation is a maximum. This means that correlation occurs at low Q, or larger scales, when H increases. Secondly, the intensity of the scattering signal is narrower in (Q,H) space than the correlation in that signal which extends over a much wider region. Finally, the intensity has a single peak in (Q,H)

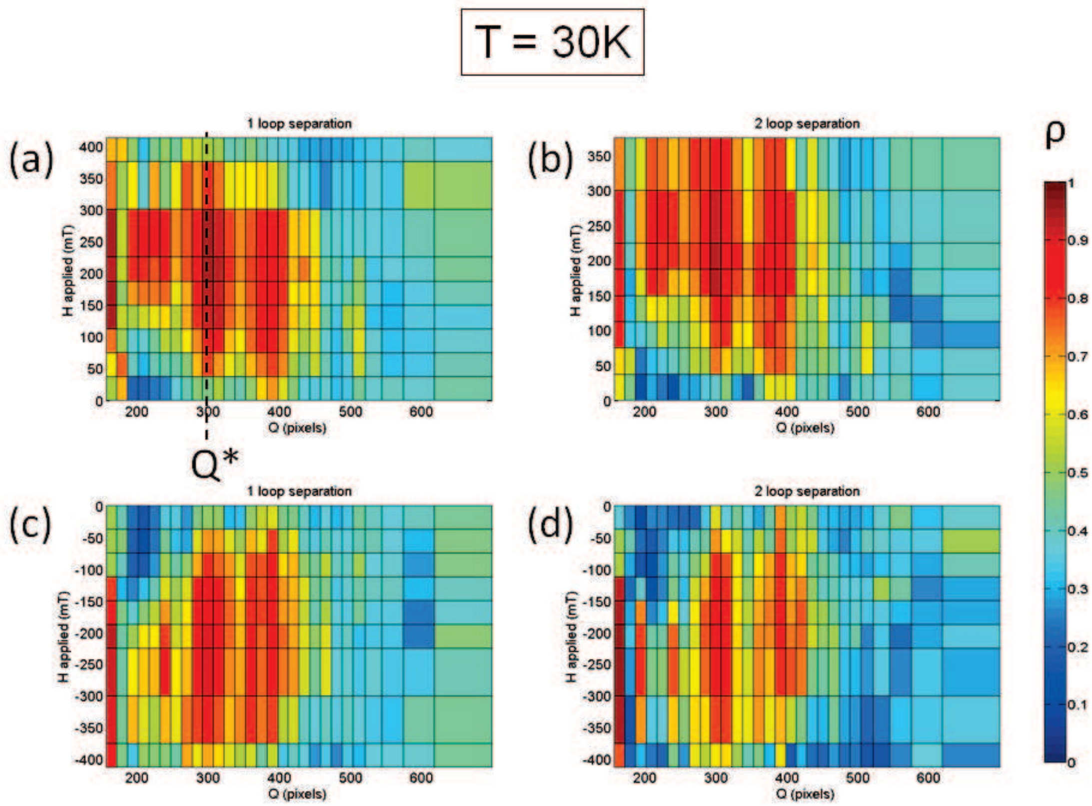

Figure 3.4 (Q,H) map at 30K (Q,H) map of correlation (RPM) as a function of field and Q at 30K. ρ is displayed on the color axis, with red representing the highest memory and blue the lowest. For each temperature, we can plot four maps, representing the correlation coefficient between images on the ascending branch (a) one loop and (b) two loops apart. Plots (c) and (d) represent the same on the descending branch of the hysteresis loop, with (c) plotting the memory at one loop separation and (d) at two loops separation. We observe a maximum in the memory at Q* = 300 pixels. Secondary peaks to the right and left of the main peak are also observed.

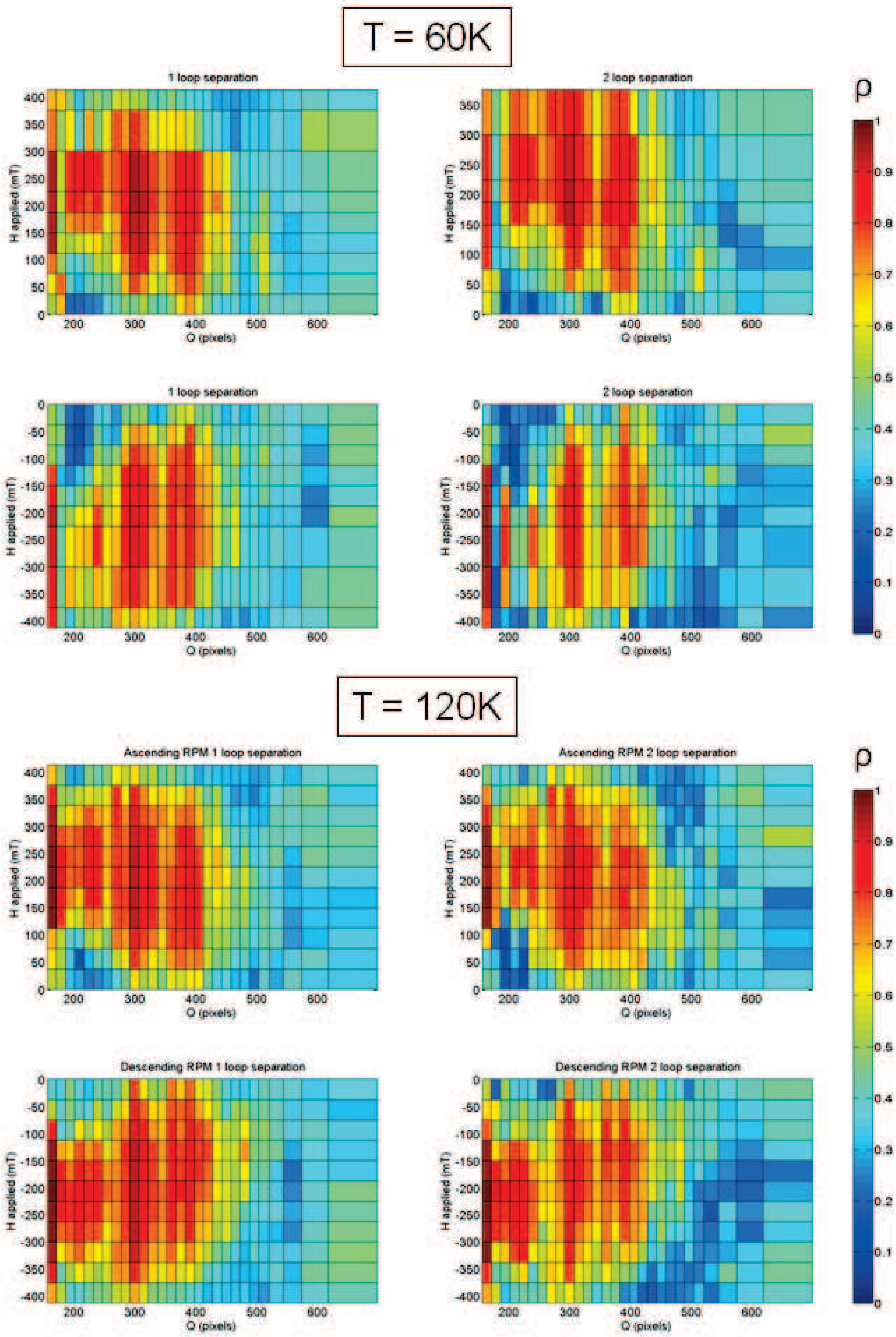

Figure 3.5 (Q,H) map at 60K, 120K (Q,H) maps of correlation (RPM) as a function of field and Q at 60K and 120K. We observe peaks in the same locations as those observed at lower temperatures.

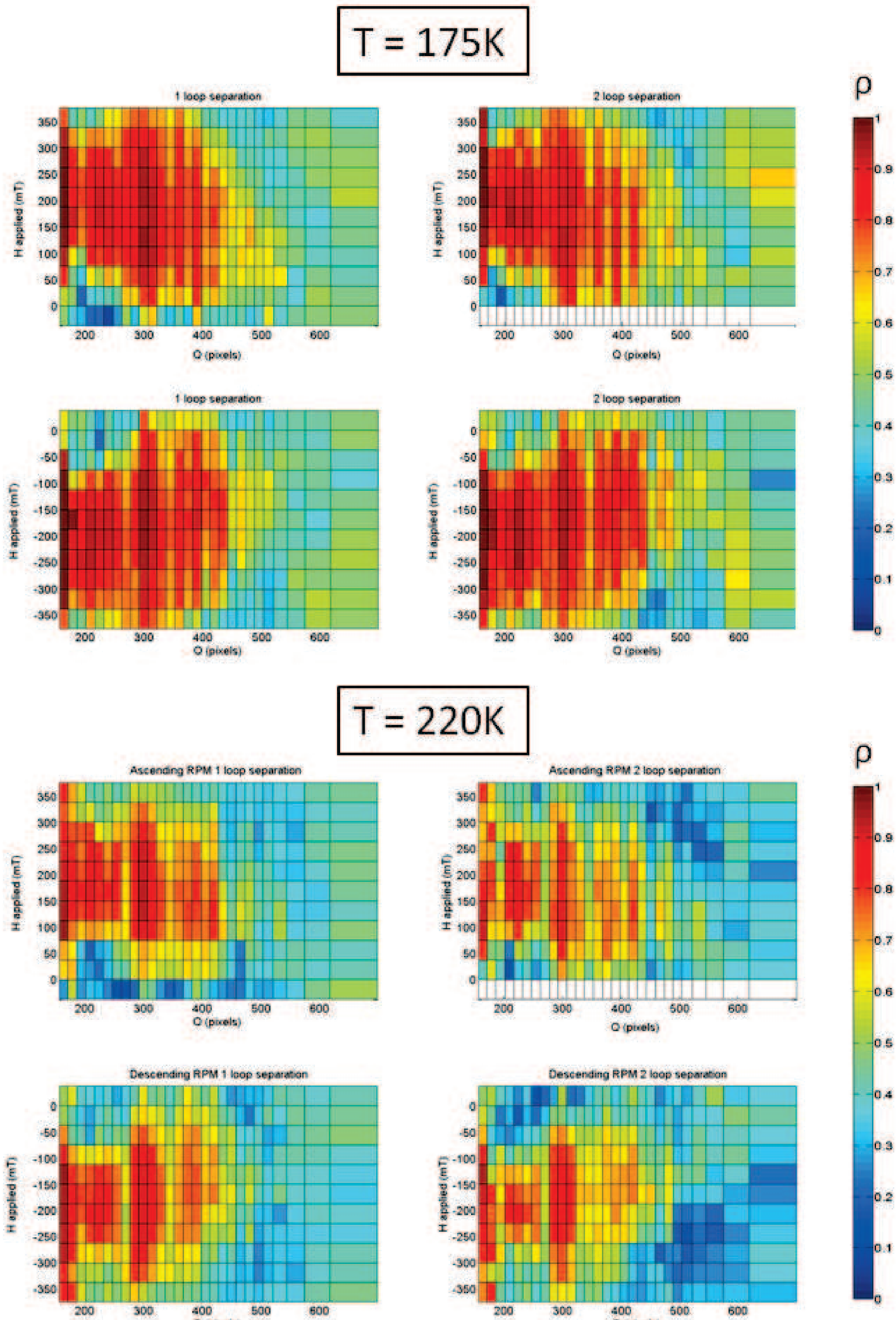

Figure 3.6 (Q,H) map at 175K, 220K (Q,H) maps of correlation (RPM) as a function of field and Q at 175K and 220K. We observe peaks in the same locations as those observed at lower temperatures, and these seem even more pronounced as we increase in temperature.

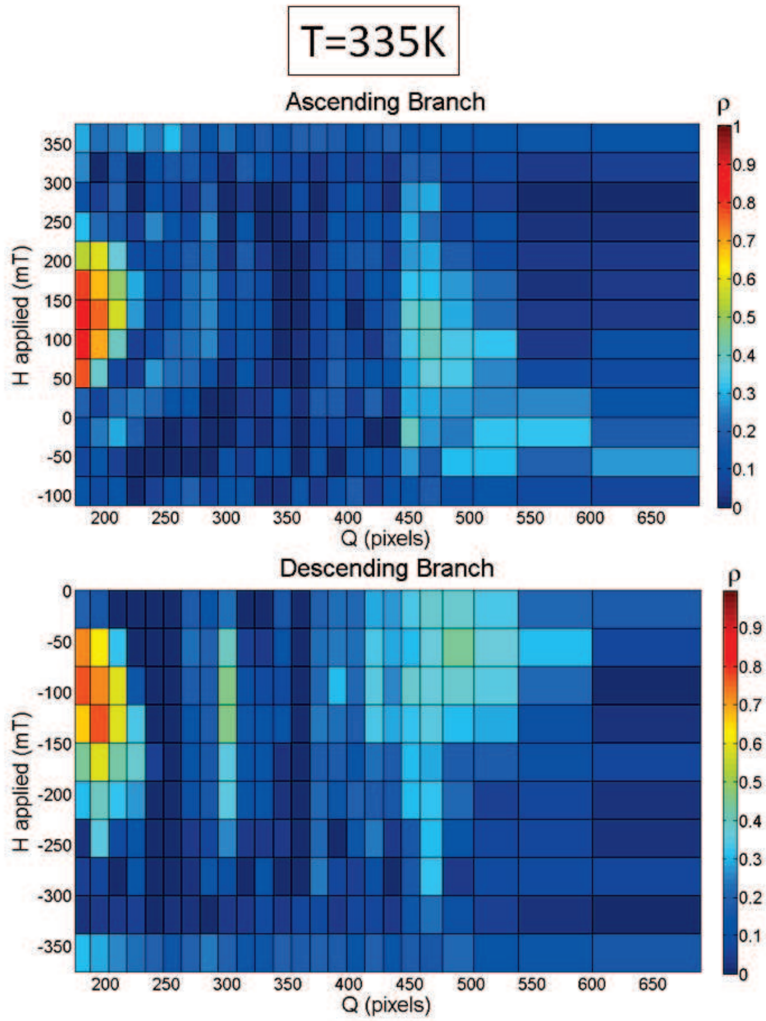

Figure 3.7 (Q,H) map at 335K (Q,H) map of correlation (RPM) as a function of field and Q at 335K. At this temperature, which is above T_B, the peaks observed at lower temperatures do not appear to be present.

space, while the correlation seems to have multiple peaks.

On the other hand, the map at high temperature, shown in Fig. 3.7, does not seem to have any of the peaks seen at lower temperature. The majority of the correlation in (Q,H) space is below 0.3. The high signal at low Q may be an artifact from leftover charge scattering near the center.

One of the conclusion we have drawn from the differences in directivity and spatial extent between the intensity and the memory is that our intensity normalization for determining ρ works correctly; the correlation between two images appears to be independent of the intensities of the images. This divergence in behavior between the intensity and memory was less obvious in "global" correlations, and has only become clear with the new information available in (Q,H) maps.

The intensity of the speckle signal as a function of Q should tell us about periodicity in the domain pattern itself. The central peak in the intensity, located at Q*, corresponds to the average domain periodicity in the sample. Thus, correlations in the ring of radius Q* can tell us about the memory of the domains at the scale of one domain period.

As we might expect, the memory is maximum at Q* in the scattering space, or at the spatial scale of an average domain period in the real space. In addition to this main peak, we also observe two clear secondary peaks in all temperatures below T_B. Interestingly, there are no peaks in the speckle intensity at these Q values. This puzzling result means that even though there is no inherent periodicity in the domains, we observe high memory at these scales. In other words, there exists a correlation in the domains at these scales without having an actual domain periodicity at these scales. If taken at face value, the location of the additional peaks indicate correlations at the scale just above and just below the average domain periodicity, but we do not observe such periodicity in the domain morphology. This has led us to search for

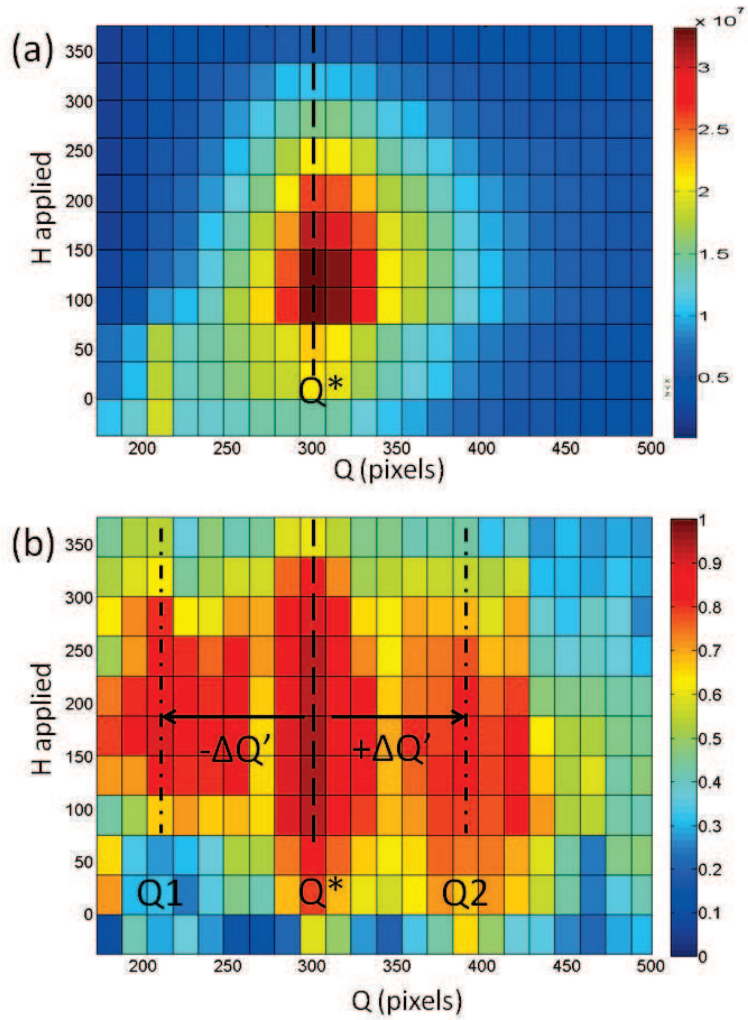

Figure 3.8 Intensity Comparison (a) Intensity of speckle signal as a function of H and Q. (b) Correlation coefficient ρ as a function of H and Q. Notably, the central peak in the correlation corresponds to the same ring at which intensity is a maximum, Q*. This ring occurs at 300 pixels, corresponding to a periodicity of about 400 nm in the real space. The central peak in the memory at Q* indicates that correlation is a maximum when performed at the same spatial scale as a domain period. Also, the secondary peaks present in the correlation, separated from the main peak by a distance of $\Delta Q'$, are not present in the intensity pattern. This strengthens the theory that these peaks result from a superstructure rather than representing two distinct periodic patterns in the domains.

alternative explanations.

Because the two secondary peaks are, within the uncertainty of our measurement, equidistant from the central peak, it suggests that rather than being actual Bragg's peaks, related to their distance from the center, these are superstructural peaks in the memory centered about the peak at Q*. Thus, the distance $\Delta Q'$ to the main peak, rather than their distances from the origin Q1 and Q2, is the indicator of the characteristic distance they correspond to in the real space.

Knowing that $\lambda = 1.59$nm, L =0.92m, and $Q*$ =285 pixels (with each pixel taking up 1.25 µm), we can conclude that the main peak at Q* corresponds to a distance in the real space of about 407.5 $\pm$ 4.5 nm. Because of the uncertainty inherent in our ring radius of 15 pixels, the uncertainty in our measurement of the spatial scale of $\Delta Q'$ is much higher. $\Delta Q'$ is measured to be 90 $\pm$ 15 pixels, which corresponds to a distance in the real space of 1.3 $\pm$ 0.2 µm. This is about three times the Q* distance, or about six domain widths.

3.3 Comparison to Magnetic Domain Images

From magnetic force microscopy (MFM) imaging of these domains, we do not see directly any repeating pattern at this 1.3 µm scale. However, the correlations between domains appears to be high at this scale. It is possible that imperfections in the deposition of the film may be influencing the shapes of the domains. In Pierce's initial work on magnetic memory, sample roughness was the only mechanism used for inducing memory [5]. However, there are no grains as large as 1.3 µm in our films. We therefore suggest that 1.3 µm may correspond to an average distance between grains in the film. This average distance would not necessarily induce any spatially repeating pattern in the domain morphology, but the grains, known to influence the shapes of domain structures in a reproducible way, are creating a secondary mechanism for memory in these samples, and thus we observe a spatial superstructure in the memory but not in the domains themselves.

To support this hypothesis, we have performed some preliminary atomic force microscopy (AFM for the remainder of the thesis refers to atomic force microscope rather than antiferromagnetic) on the film used in this scattering to try to determine the average spacing between grains. The atomic force microscope image as well as its MFM counterpart are shown in Fig. 3.9. Because the AFM details the surface of the sample, it is possible that the features imaged here are not grains within the sample, but debris on its outer surface. We plan to conduct much more AFM imaging to get a better idea about the structure within the sample. The MFM image gives us a great deal of information, however. We can observe an average periodicity of about 415nm, with a very short correlation length. This indicates what is going on in the ZFC state.

3.4 Future Work

We suggest a study to measure the memory of exchange-bias films with a controlled amount of structural grains and analyze these data using our radius-selective procedure. Memory was very low at room temperature, so our memory is obviously not dominated by defects. If our hypothesis is correct, the distance between the grains will decrease as surface roughness increases, and therefore the distance between peaks in Q-space should increase. This would confirm that the additional peaks in the memory are caused by structural defects in the film influencing the shapes of domains in a reproducible way, creating a secondary mechanism for memory in these films.

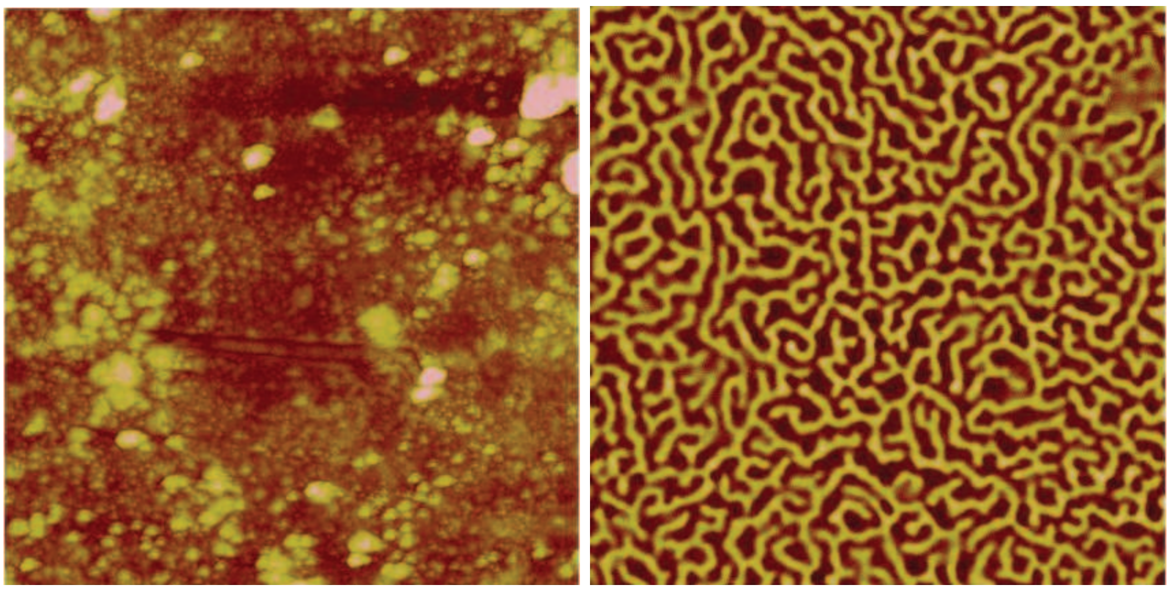

Figure 3.9 AFM/MFM Comparison (Left) Atomic Force Microscope (AFM) image of the exchange-bias thin film used in this study. Image is 10 µm squared, with a vertical colorscale of 100 nm. The features imaged here are a preliminary check to try to measure the average distances between defects. However, these features may be debris on the surface of the sample. (Right) Magnetic Force Microscope (MFM) image of the same region of the film. The disordered magnetic domains do not show obvious morphological dependence on any of the features imaged in the AFM. They do, however, show a periodicity consistent with that measured from the scattering.

3.5 Conclusion

We have studied magnetic domain memory as a function of field, temperature, and spatial scale. We have observed that domain memory exhibits some unique spatial features in Q that do not mimic the behavior of the scattering intensity in (Q,H) space. This suggests that the magnetic domain memory extends over a very wide spatial scale going from down below the domain periodicity to well above it and even being intensified at a superstructural scale including about 7 domain periods.

This result is very uniquely obtained by this Q-selective cross-correlation technique. We also observe this behavior to be quite robust with increasing temperature all the way up to the T_B.

Bibliography

[1] K. Chesnel, E. E. Fullerton, M. J. Carey, J. B. Kortright, and S. D. Kevan, "Magnetic memory in ferromagnetic thin films via exchange coupling," Phys. Rev. B **78,** 132409 (2008).

[2] H. Richter, "Density limits imposed by the microstructure of magnetic recording media," Journal of Magnetism and Magnetic Materials **321,** 467 – 476 (2009), current Perspectives: Perpendicular Recording.

[3] S. Hashimoto, Y. Ochiai, and K. Aso, "Perpendicular magnetic anisotropy and magnetostriction of sputtered Co/Pd and Co/Pt multilayered films," Journal of Applied Physics **66,** 4909–4916 (1989).

[4] C. Kittel, "Physical Theory of Ferromagnetic Domains," Rev. Mod. Phys. **21,** 541–583 (1949).

[5] M. S. Pierce *et al.*, "Disorder-induced magnetic memory: Experiments and theories," Phys. Rev. B **75,** 144406 (2007).

[6] J. Nogues and I. Schuller, "Exchange bias," J. Mag. Mag. Mat. **192,** 203–232 (1999).

[7] D. Mauri, E. Kay, D. Scholl, and J. Howard, "Novel method for determining the anisotropy constant of MnFe in a NiFe/MnFe sandwich," J. Appl. Phys. **62,** 2929–2932 (1987).

[8] W. H. Meiklejohn and C. P. Bean, "New Magnetic Anisotropy," Phys. Rev. **105,** 904–913 (1957).

[9] S. Maat, K. Takano, S. Parkin, and E. Fullerton, "Perpendicular exchange bias of Co/Pt multilayers," Phys. Rev. Lett. **87,** 087202 (2001).

[10] J. Eckert, N. Stern, A. Barton, D. Mann, P. Sparks, and M. Carey, "Low temperature properties of spin valves with extremely thin IrMn," J. Appl. Phys. **91,** 8569–8571 (2002).

[11] J. Kortright, D. Awschalom, J. Stohr, S. Bader, Y. Idzerda, S. Parkin, I. Schuller, and H. Siegmann, "Research frontiers in magnetic materials at soft X-ray synchrotron radiation facilities," J. Mag. Mag. Mat. **207,** 7–44 (1999).

[12] K. Chesnel, J. J. Turner, M. Pfeifer, and S. D. Kevan, "Probing complex materials with coherent soft X-rays," Applied Physics A-Materials Science and Processing **92,** 431–437 (2008).

[13] S. Eisebitt, M. Lorgen, W. Eberhardt, J. Luning, J. Stohr, C. Rettner, O. Hellwig, E. Fullerton, and G. Denbeaux, "Polarization effects in coherent scattering from magnetic specimen: Implications for x-ray holography, lensless imaging, and correlation spectroscopy," Phys. Rev. B **68,** 104419 (2003).

[14] K. Chesnel, M. Belakhovsky, F. Livet, S. Collins, G. van der Laan, S. Dhesi, J. Attane, and A. Marty, "Soft-x-ray magnetic speckles from a nanostructured FePd wire," Phys. Rev. B **66,** 172404 (2002).

[15] B. Wilcken, "Magnetic Memory in Exchange-Bias Thin Films,", Senior Thesis, Brigham Young University, 2009.

[16] M. Pierce, R. Moore, L. Sorensen, S. Kevan, O. Hellwig, E. Fullerton, and J. Kortright, "Quasistatic x-ray speckle metrology of microscopic magnetic return-point memory," Phys. Rev. Lett. **90,** 017202 (2003).

Appendix A

Code

A.1 Speckle Isolation Code

Two of the most basic programs used were getTetragon and getEllipse. These return a matrix of a given size with an ellipse or tetragon filled with ones while the rest of the image is zeros (or vice-versa). Their code is:

getTetragon.m

```
function tetragon = getTetragon(x1,y1,x2,y2,x3,y3,x4,y4,matrixSize,invert)
    %Line eqaution: y = y1 + (y2 - y1)/(x2 - x1)*(x - x1)
    if (x1 == x2)
        x2 = x2 + 0.001;
    end
    if (x2 == x3)
        x3 = x3 + 0.001;
    end
    if (x3 == x4)
        x4 = x4 + 0.001;
    end
    if (x1 == x4)
        x4 = x4 + 0.001;
    end

    N = 10000;
    stepA = (x2 - x1)/N;
    stepB = (x3 - x2)/N;
    stepC = (x4 - x3)/N;
    stepD = (x4 - x1)/N;
    xA = x1:stepA:x2;
    xB = x2:stepB:x3;
```

```
    xC = x3:stepC:x4;
    xD = x1:stepD:x4;
    yA = y1 + (y2 - y1)/(x2 - x1)*(xA - x1);
    yB = y2 + (y3 - y2)/(x3 - x2)*(xB - x2);
    yC = y3 + (y4 - y3)/(x4 - x3)*(xC - x3);
    yD = y1 + (y4 - y1)/(x4 - x1)*(xD - x1);
    for n = 1:N
        xA(n) = fix(xA(n));
        xB(n) = fix(xB(n));
        xC(n) = fix(xC(n));
        xD(n) = fix(xD(n));
        yA(n) = fix(yA(n));
        yB(n) = fix(yB(n));
        yC(n) = fix(yC(n));
        yD(n) = fix(yD(n));
    end
    tetragon = zeros(matrixSize);
    for n = 1:N
        tetragon(yA(n),xA(n)) = 1;
        tetragon(yB(n),xB(n)) = 1;
        tetragon(yC(n),xC(n)) = 1;
        tetragon(yD(n),xD(n)) = 1;
    end
    tetragonL = tetragon;
    tetragonR = tetragon;
    for m = 1:matrixSize(1) %rows
        rowMarked = 0;
        for n = 1:matrixSize(2) %columns
            if (tetragonL(m,n) == 1 && rowMarked)
                continue;
            end
            if (tetragonL(m,n) == 1 && ~rowMarked)
                rowMarked = 1;
            end
            if (tetragonL(m,n) == 0 && rowMarked)
                tetragonL(m,n) = 1;
            end
        end
    end
    for m = matrixSize(1):-1:1 %rows
        rowMarked = 0;
        for n = matrixSize(2):-1:1 %columns
            if (tetragonR(m,n) == 1 && rowMarked)
```

```
                continue;
            end
            if (tetragonR(m,n) == 1 && ~rowMarked)
                rowMarked = 1;
            end
            if (tetragonR(m,n) == 0 && rowMarked)
                tetragonR(m,n) = 1;
            end
        end
    end
    tetragon = tetragonR.*tetragonL;
    if (invert)
        tetragon = ones(matrixSize) - tetragon;
    end
end
```

and getEllipse2, which is the same as getEllipse, but the ellipse can extend beyond the scope of the image.

getEllipse2

```
function ellipse = getEllipse2(h,k,a,b,phi,matrixSize,invert)
    N = 10000;
    start = -pi;
    stop = pi;
    step = (stop - start)/N;
    theta = -pi:step:pi;
    x = k + a*cos(theta)*cos(phi) - b*sin(theta)*sin(phi);
    y = h + b*sin(theta)*cos(phi) + a*cos(theta)*sin(phi);
    for q = 1:length(x)
        x(q) = round(x(q));
        if x(q)<1
            x(q)=1;
        end
        if x(q)>matrixSize(2)
            x(q)=matrixSize(2);
        end
        y(q) = round(y(q));
        if y(q)<1
            y(q)=1;
        end
        if y(q)>matrixSize(1)
            y(q)=matrixSize(1);
        end
    end
```

```

    ellipse = zeros(matrixSize);
    for n = 1:length(x)
        ellipse(x(n),y(n)) = 1;
    end
    ellipseL = ellipse;
    ellipseR = ellipse;
    for m = 1:matrixSize(1) %rows
        rowMarked = 0;
        for n = 1:matrixSize(2) %columns
            if (ellipseL(m,n) == 1 && rowMarked)
                continue;
            end
            if (ellipseL(m,n) == 1 && ~rowMarked)
                rowMarked = 1;
            end
            if (ellipseL(m,n) == 0 && rowMarked)
                ellipseL(m,n) = 1;
            end
        end
    end
    for m = matrixSize(1):-1:1 %rows
        rowMarked = 0;
        for n = matrixSize(2):-1:1 %columns
            if (ellipseR(m,n) == 1 && rowMarked)
                continue;
            end
            if (ellipseR(m,n) == 1 && ~rowMarked)
                rowMarked = 1;
            end
            if (ellipseR(m,n) == 0 && rowMarked)
                ellipseR(m,n) = 1;
            end
        end
    end
    ellipse = ellipseR.*ellipseL;
    if (invert)
        ellipse = ones(matrixSize) - ellipse;
    end
end
```

Here is an example of a file for reading in the images, storing them in cell array Im, and performing the speckle isolation process (in the function getSpeckle2), and storing these pure speckle images in the cell array Imspeck.

SpeckleMaker3200.m

```
% This file declares the images and calculates and saves the speckle images for the 3100 series.

% IMPORTANT: The cell Im's first 3 rows are the ascending branches, and
% final 3 rows are the descending branches

ref{1}=fitsread('eb3187.fit');
ref{2}=fitsread('eb3213.fit');
ref{3}=fitsread('eb3239.fit');
ref{4}=fitsread('eb3200.fit');
ref{5}=fitsread('eb3226.fit');
ref{6}=fitsread('eb3252.fit');

S=size(ref{1});

load('ascend3200.mat')
load('descend3200.mat')

Im{6,12}=[];
for n1=1:3
    for n2=1:12
        if ascend(n2,n1) ~= 0
        Im{n1,n2} = eval(['fitsread(' '''' 'eb' int2str(ascend(n2,n1)) '.fit' '''' ');'])-ref{n1};
        end
        if descend(n2,n1) ~= 0
        Im{n1+3,n2} = eval(['fitsread(' '''' 'eb' int2str(descend(n2,n1)) '.fit' '''' ');'])-ref{n1+3};
        end
    end
end

tetra=getTetragon(671,419,1001,445,1001,570,636,542,[1001,1001],1);
elli=getEllipse(498,465,180,173,.48*pi,[1001,1001],1);

Imspeck{6,12}=[];
```

```
for T1=1:6
    for T2=1:12
        if isempty(Im{T1,T2}) == 0
            Imspeck{T1,T2}=getSpeckle2(Im{T1,T2},tetra,elli);
        end
    end
end

save Speckle3200_double DMspeck Imspeck
save Im3200_2 Im
```

The function doubleboundaryfit is a 1-D polynomial fit of the columns of the images, followed by a 1-D fit of the rows of the image (now with the fitted column values inserted where the blocker was). The double-fitted values then replace the regions specified to be behind the blocker (These are specified by the ellipse and tetragon elli and tetra, which are simply zeroed-out regions where the blocker is located).

doubleboundaryfit.m

```
function cyborg2 = doubleboundaryfit(im,blocker,deg2)
deg1=1;
S=size(im);
zerolim=3;
im=im.*blocker;
cyborg=im;
Z=1-blocker;
for n=1:S(2)
    z=Z(:,n);
    col=im(:,n);
    zback=rot90(z,2);
    if sum(z) > zerolim
        [a,i1]=max(z);
        [a,i2]=max(zback);
        i1=i1-1;
        i2=S(2)-i2+2;
        x=[i1-3:i1,i2:i2+3];
        y=col(x);
        full=1:S(2);
        p=quietpolyfit(x',y,deg1);
        filler=polyval(p,full);
        cyborg(:,n)=filler'.*z+col;
    end
end
full=1:S(1);
cyborg2=cyborg;
for n=1:S(1)
    z=Z(n,:);
    row=cyborg(n,:);
    if sum(z) > zerolim
        p=quietpolyfit(full,row,deg2);
        filler=polyval(p,full);
        cyborg2(n,:)=filler.*z+im(n,:);
    end
end
```

The function getSpeckle2 is an improvement of Brian Wilcken's code getSpeckle. The only real difference is this program utilizes the function doubleboundaryfit to fit the region behind the blocker prior to the smoothing process to eliminate the boundary problem.

getSpeckle2.m

```
function [speckle,blurredImage] = getSpeckle2(image,ellipse,tetragon)
    vis = 0; %1 = display graphs, 0 = no display
    blockerRemove = ellipse.*tetragon;
    dTolFinder = image.*blockerRemove;
    dTol = (std(std(dTolFinder))/(max(max(dTolFinder))-min(min(dTolFinder))))/2;
    %fprintf('Differential Tolerance: %f\n',dTol)
    PSF = fspecial('average',3); %3x3 point spread function (PSF) used in convolution for image blurring
    blurredImage = doubleboundaryfit(image,blockerRemove,15);
    p = 1; %counter for generating range info two steps ahead
    n = 3; %counter for calculating centered derivatives
    previousOne = 1;
    current = 1;
    blurredImageOne = 1;
    currentBlur = 1;
    while(true)
        blurredImage = imfilter(blurredImage,PSF,'conv','replicate'); %FFT2 based image blurring via convolution
%           blurredImage = PbPBlur(blurredImage); %Pixel-by-pixel convolution based blurring
        speckle = (image - blurredImage).*blockerRemove; %Remove blocker
        %speckle = ImageCrop(speckle); %crop a percentage of pixels from each side of the speckle, so as to discard with edge effects inherent in the blurring techniques
        maximum(p) = max(max(speckle));
        minimum(p) = min(min(speckle));
        r(p) = maximum(p) - minimum(p); %r is a vector of length 1:p that contains the range of the speckle at any given pass p: r(p)
        blurMax(p) = max(max(blurredImage));
        blurMin(p) = min(min(blurredImage));
        blurR(p) = blurMax(p) - blurMin(p); %record the range
```

```
of the blur to show how it decays over time
        %Succesive Substitutions for returning the correct speckle result
        previousTwo = previousOne;
        previousOne = current;
        current = speckle;
        blurredImageTwo = blurredImageOne;
        blurredImageOne = currentBlur;
        currentBlur = blurredImage;
        if (p > 4) %the 1st and 2nd centered difference derivatives cannot be defined without a minimum of 3 data points
            drdp(n) = (r(n+1) - r(n-1))/(2*1); %centered difference 1st p derivative of r(p)
            d2rdp2(n) = (r(n+1) - 2*r(n) + r(n-1))/(1^2); %centered difference 2nd p derivative of r(p)
            d3rdp3(n) = (r(n+2) - 2*(r(n+1) - r(n-1)) - r(n-2))/(2*1^3); %centered 3rd p derivative of r(p)
            if (vis) %these lines of code only execute if vis was not set to 0, the following code presents an array of graphs and images in a useful layout that is easy to read
                %setup speckle slice visualization
                [vS,hS] = size(speckle);
                original = image.*blockerRemove;
                origSlice = original(:,ceil(hS/2));
                blurSlice = blurredImageTwo.*blockerRemove;
                blurSlice = blurSlice(:,ceil(hS/2));
                speckleSlice = previousTwo(:,ceil(hS/2));
                %Blur evolution
                subplot(4,6,[1 2 7 8]);
                imagesc(blurredImageTwo)
                info = sprintf('blur after %d passes',n);
                title(info)
                %Speckle evolution (image)
                subplot(4,6,[3 4 9 10]);
                imagesc(previousTwo)
                info = sprintf('speckle after %d passes',n);
                title(info)
                %Speckle evolution (slice)
                subplot(4,6,[5 6 11 12]);
                plot(origSlice,'y-')
```

```
        hold on
        plot(speckleSlice,'r-')
        plot(blurSlice,'k-')
        plot(0*blurSlice,'m-')
        hold off
        info = sprintf('Vertical Slice\nYellow: original, Black: blur, Red: speckle');
        title(info)
        %AC result
        subplot(4,6,[17 18 23 24]);
        surf(CyclicFFT2xcorr(previousTwo,previousTwo))
        info = sprintf('auto correlation result after %d passes',n);
        title(info)
        %plot of speckle range
        subplot(4,6,[15 16]);
        plot(r(1:n))
        hold on
        plot(maximum(1:n),'k-')
        plot(minimum(1:n),'r-')
        hold off
        info = sprintf('Speckle Range (r)\nr = %f',r(n));
        title(info)
        %plot of blur range
        subplot(4,6,[13 14]);
        plot(blurR(1:n),'b-')
        hold on
        plot(blurMax(1:n),'k-')
        plot(blurMin(1:n),'r-')
        hold off
        info = sprintf('range of blur = %f',blurR(n));
        title(info)
        %plot of 1st derivative of speckle range as a function of
        %pass
        subplot(4,6,19);
        plot(drdp)
        info = sprintf('1st derivative of range at %d passes\ndr/dp = %f',n,drdp(n));
        title(info)
```

```
                %plot of 2nd derivative of speckle range as a function of
                %pass
                subplot(4,6,21:22);
                plot(d2rdp2)
                info = sprintf('2nd derivative of range at %d passes\nd2r/dp2 = %f',n,d2rdp2(n));
                title(info)
                %plot of 3rd derivative of speckle range as a function of
                %pass
                subplot(4,6,20);
                plot(d3rdp3)
                info = sprintf('3rd derivative of range at %d passes\nd3r/dp3 = %f',n,d3rdp3(n));
                title(info)
                pause(0.01)
            end
            if (abs(d2rdp2(n)) < dTol && abs(d3rdp3(n)) < abs(d2rdp2(n))) %break condition for the while loop executes if we have achieved convergence of the centered 2nd derivative of r(p) to within tolerance
                break
            end
            n = n + 1;
        end
        p = p + 1;
    end
    speckle = previousTwo;
    %fprintf('Speckle range(iterations) 2nd derivative convergence to < %f in %d iterations.\n',dTol,p)
end
```

A.2 Center Fitting

This is the program I used for fitting the center of the (slightly elliptical) ring in one of the image series. It returns the location of the center h,k as well as the ratio of the semi-major and semi-minor axes btoa, and the angle of the ellipse phi.

FitEllipseCenter.m

```
function [h,k,a,b,phi,btoa]=FitEllipseCenter(im)
%this function takes as input the blurred image (central
    region need not be
%removed).

S=fix(size(im)/10);
for i=1:3
     vect(i)=(3+2*(i-1))*S(1);
     x(i,:)=im(vect(i),:);
     y(i,:)=im(:,vect(i))';
end

d=diag(im)';

M=fix(length(im)/4);
first(1:2*M(1))=1;
first(2*M(1):length(x))=0;
last(1:2*M(1))=0;
last(2*M(1):length(x))=1;
for i=1:3
     [C,xval(1,i)]=max(x(i,:).*first);
     [C,xval(2,i)]=max(x(i,:).*last);
     [C,yval(1,i)]=max(y(i,:).*first);
     [C,yval(2,i)]=max(y(i,:).*last);
end
[C,dval(1)]=max(d.*first);
[C,dval(2)]=max(d.*last);

XY(1,:)=[yval(1,1),vect(1)];
XY(2,:)=[yval(2,1),vect(1)];
XY(3,:)=[yval(1,2),vect(2)];
XY(4,:)=[yval(2,2),vect(2)];
XY(5,:)=[yval(1,3),vect(3)];
XY(6,:)=[yval(2,3),vect(3)];
XY(7,:)=[vect(1),xval(1,1)];
XY(8,:)=[vect(1),xval(2,1)];
XY(9,:)=[vect(2),xval(1,2)];
```

```
% XY(8,:)=[vect(2),xval(2,2)];
XY(10,:)=[vect(3),xval(1,3)];
XY(11,:)=[vect(3),xval(2,3)];
XY(12,:)=[dval(1),dval(1)];
XY(13,:)=[dval(2),dval(2)];

% whitespace=ones(size(im));
% for i=1:length(XY)
%       whitespace(XY(i,1)-3:XY(i,1)+3,XY(i,2)-3:XY(i,2)+3)=0;
% %       imagesc(whitespace.*im)
% %       colormap gray
% %       pause
% end

[a,b,k,h,phi]=ellipse_fit(XY(:,1),XY(:,2));
btoa=b/a;


% e1=getEllipse(h,k,a,b,phi,size(im),0);
% e2=getEllipse(h,k,a+5,b+5,phi,size(im),1);
% E=e1+e2;
%
% % B=CircleFit(XY);
% % mid=[round(B(1)),round(B(2))];
% % r=round(B(3));
% % e3=getEllipse(mid(1),mid(2),r,r,0,size(im),0);
% % e4=getEllipse(mid(1),mid(2),r+5,r+5,0,size(im),1);
% % C=e3+e4;
% % subplot(1,2,1)
% imagesc(im.*E.*whitespace); title('Elliptical fit')
% % subplot(1,2,2); imagesc(im.*C.*whitespace); title('Circular
    fit')
% % colormap gray
% pause(.1)
```

I used images in the coercive region, and averaged the output of FitEllipseCenter.m to find the value of the center.

CenterSurvey3200.m

```
load('Im3200.mat')
hvalues=zeros(5,5);
kvalues=zeros(5,5);
ratios=zeros(5,5);
phivalues=zeros(5,5);
```

```

for T1=1:6
    i=1;
    for T2=3:7
        if isempty(Im{T1,T2}) == 0
        im=overblur(Im{T1,T2});
        [h,k,a,b,phi,btoa]=FitEllipseCenter(im);
        hvalues(T1,i)=h;
        kvalues(T1,i)=k;
        ratios(T1,i)=btoa;
        phivalues(T1,i)=phi;
        i=i+1;
        end
    end
end

numzeros=sum(sum(isinf(1./hvalues)));
s=size(hvalues);
div=s(1)*s(2)-numzeros;

h=sum(sum(hvalues))/div;
k=sum(sum(kvalues))/div;
btoa=sum(sum(ratios))/div;
phi=sum(sum(phivalues))/div;

save CenterEllipse3200.mat h k btoa phi
```

A.3 Cross-Correlation

Here is an example of a whole-image cross-correlation algorithm. rhoA and rhoD are cell arrays, with each cell corresponding to the number of loops separating the correlated images.

nonq3200_rpm.m

```
clear;
load('Speckle2900.mat');

S=size(Imspeck{1,5});
mid=floor(S(1)/2);

%ellipse over which the auto/cross-correlation peaks are integrated
s=getEllipse(round(S(1)/2),round(S(2)/2),25,12,(.48*pi),S,0);

% Autocorrelation
[W,L]=size(Imspeck);

Auto(W,L)=0;
for T1=1:W
    for T2=1:L
        if isempty(Imspeck{T1,T2}) == 0
            C=ifft2(fft2(Imspeck{T1,T2}).*fft2(rot90(Imspeck{T1,T2},2)));
            C=fftshift(C);
            Auto(T1,T2)=sum(sum(s.*(C+abs(C))/2));
        end
    end
end

% Cross-correlation
loops=W/2-1;

rhoA{loops}=[];
rhoD{loops}=[];
counterA=zeros(L,loops);
counterD=zeros(L,loops);
for T1=1:W/2
    T=T1+W/2;
    for m=1:W/2-T1
        rhoA{m}(L)=0;
        rhoD{m}(L)=0;
```

```
            for T2=1:L
                if isempty(Imspeck{T1,T2}) == 0
                    if isempty(Imspeck{T1+m,T2}) == 0
                        C=ifft2(fft2(Imspeck{T1,T2}).*fft2(rot90(
                            Imspeck{T1+m,T2},2)));
                        C=fftshift(C);
                        surf(C(mid-50:mid+50,mid-50:mid+50));
                        pause
                        rhoA{m}(T2)=rhoA{m}(T2)+sum(sum(s.*(C+abs
                            (C))/2))/sqrt(Auto(T1,T2)*Auto(T1+m,T2
                            ));
                        counterA(T2,m)=counterA(T2,m)+1;
                    end
                end
                if isempty(Imspeck{T,T2}) == 0
                    if isempty(Imspeck{T+m,T2}) == 0
                        C=ifft2(fft2(Imspeck{T,T2}).*fft2(rot90(
                            Imspeck{T+m,T2},2)));
                        C=fftshift(C);
                        rhoD{m}(T2)=rhoD{m}(T2)+sum(sum(s.*(C+abs
                            (C))/2))/sqrt(Auto(T,T2)*Auto(T+m,T2))
                            ;
                        counterD(T2,m)=counterD(T2,m)+1;
                    end
                end
            end
        end
end
for m=1:loops
    for T2=1:L
        if counterA(T2,m) ~= 0
            rhoA{m}(T2)=rhoA{m}(T2)/counterA(T2,m);
        end
        if counterD(T2,m) ~= 0
            rhoD{m}(T2)=rhoD{m}(T2)/counterD(T2,m);
        end
    end
end

save not-q_2900_rpm_double_results.mat rhoA rhoD

% Remember: the first 3 rows of rhoA/D are ascending, last 3
    are descending
```

Here is the same series correlation, but with Q-selective correlations. It employs the nearest-neighbor fit in the autocorrelation.

q3200_rpm_cw.m

```
load('Speckle3200_double.mat');
load('CenterEllipse3200.mat');

S=size(Imspeck{1,5});

% load the radius values for the rings
load('cw15_1000_radius.mat')

% Make the rings
qold=getEllipse2(h,k,r(1),r(1)*btoa,phi,S,1);
for n=2:length(r)
    qnew = getEllipse2(h,k,r(n),r(n)*btoa,phi,S,1);
    Q{n-1} = qold-qnew;
    pix(n-1)=sum(sum(Q{n-1}));
    if pix(n-1) == 0
        break
    end
    qold = qnew;
end

L=length(Q);

% Ellipse over which auto/cross-correlation peaks are
    integrated
s=getEllipse(round(S(1)/2),round(S(2)/2),25,12,(.48*pi),S,0);

% Autocorrelation
AutoDM(L)=0;
for n=1:L
    C=ifft2(fft2(DMspeck.*Q{n}).*fft2(rot90(DMspeck.*Q{n},2))
        );
    C=fftshift(C);
    AutoDM(n)=sum(sum(s.*(C+abs(C))/2));
end

Auto{6,12}=[];
for T1=1:6
    for T2=1:12
        Auto{T1,T2}(L)=0;
```

```
            for n=1:L
                if isempty(Imspeck{T1,T2}) == 0
                    C=ifft2(fft2(Imspeck{T1,T2}.*Q{n}).*fft2(rot90(Imspeck{T1,T2}.*Q{n},2)));
                    C=fftshift(C);
                    [Val1,I1]=max(C);
                    [zdp,I2]=max(Val1);
                    C(I1(I2),I2)=0;
                    neighbor=max(max(C));
                    C(I1(I2),I2)=neighbor;
                    Auto{T1,T2}(n)=sum(sum(s.*(C+abs(C))/2));
                end
            end
        end
end

% Cross Correlation (RPM)

rhoA{2}=[];
rhoD{2}=[];
counterA=zeros(12,2);
counterD=zeros(12,2);
for T1=1:3
    T=T1+3;
    for m=1:3-T1
        rhoA{m}(12,L)=0;
        rhoD{m}(12,L)=0;
        for T2=1:12
            if isempty(Imspeck{T1,T2}) == 0
                if isempty(Imspeck{T1+m,T2}) == 0
                    for n=1:L
                        C=ifft2(fft2(Imspeck{T1,T2}.*Q{n}).*fft2(rot90(Imspeck{T1+m,T2}.*Q{n},2)));
                        C=fftshift(C);
                        rhoA{m}(T2,n)=rhoA{m}(T2,n)+sum(sum(s.*(C+abs(C))/2))/sqrt(Auto{T1,T2}(n)*Auto{T1+m,T2}(n));
                    end
                    counterA(T2,m)=counterA(T2,m)+1;
                end
            end
            if isempty(Imspeck{T,T2}) == 0
```

```
                    if isempty(Imspeck{T+m,T2}) == 0
                        for n=1:L
                            C=ifft2(fft2(Imspeck{T,T2}.*Q{n}).*
                                fft2(rot90(Imspeck{T+m,T2}.*Q{n
                                },2)));
                            C=fftshift(C);
                            rhoD{m}(T2,n)=rhoD{m}(T2,n)+sum(sum(s
                                .*(C+abs(C))/2))/sqrt(Auto{T,T2}(n
                                )*Auto{T+m,T2}(n));
                        end
                        counterD(T2,m)=counterD(T2,m)+1;
                    end
                end
            end
        end
end
for m=1:2
    for T2=1:12
        if counterA(T2,m) ~= 0
            rhoA{m}(T2,:)=rhoA{m}(T2,:)/counterA(T2,m);
        end
        if counterD(T2,m) ~= 0
            rhoD{m}(T2,:)=rhoD{m}(T2,:)/counterD(T2,m);
        end
    end
end
save q3200_rpm_cw_neighbor_results.mat rhoA rhoD pix

% Remember: the first three rows of rho are ascending, last
    three are
% descending.
```